Preface

Declarative programming languages provide a high-level style of programming which reduces program development time and improves the maintainability and verifiability of programs in comparison with more conventional methods using imperative languages. Currently, declarative programming is split into two main fields, namely, functional programming and logic programming. However, in spite of the fact that these fields are very closely related in many ways, there has been little interaction between them. Few researchers are involved in both fields and each field has its own journals, conferences, and research agenda which have developed largely in isolation from one another.

Over 15 years ago, a number of people tried to overcome this situation and started research related to the problem of integrating functional and logic languages which has continued to this day. Recently, it has been shown that functional logic languages can be practically useful since the execution principles of functional logic languages have been refined in order to avoid many non-deterministic choices and the implementation of declarative languages has been improved considerably. Currently, there is active research in the following topics:

- The design of new programming languages accounting for a mature and up-to-date understanding of previously studied features meant to improve the expressive power of declarative languages (lazy evaluation, higher-order objects, types, object orientation, etc...).

- The development of efficient implementations. The proposals combine existing techniques from logic programming (e.g., WAM, mode analysis, abstract interpretation) and functional programming (e.g., graph reduction, strictness analysis, optimization of deterministic computations) as well as new techniques (e.g., combination of normalization and narrowing, residuation and coroutining, parallelization).

- Integration with other paradigms, such as the inclusion of constraints, object-oriented programming, concurrent programming, etc.

Many results of recent work on the integration of functional and logic languages have been presented at other specialized conferences (e.g., ALP, FPCA, PLILP, RTA). The aim of this workshop was to bring together researchers from different communities and to stimulate discussions. Another important aim was to discuss the design of a language which integrates the most important features of modern functional and logic programming languages. Although various integrated functional logic languages have been proposed in the past, the individual computation models are very different. Thus, the participants tried to find an agreement on a language

which can serve as a standard platform to exchange applications and to coordinate future developments.

In order to structure the discussion on such future developments, the workshop started with introductory talks about general topics and approaches to integrate the functional and logic paradigms. On the second day, smaller working groups were formed to try to identify challenges and features of integrated languages w.r.t. to different aspects (semantics, application, implementation). These working groups met during the entire week, i.e., technical talks were held in the morning and the afternoons (and sometimes the evenings) were reserved for discussion in the working groups. On the last day, we had a plenary discussion about the results of the working groups. During this last discussion, it became clear that this workshop reached its goals. Due to ample time for discussions, the working groups have identified the basic features of a future language where everybody agreed upon. Although the details must be worked out in the future, this workshop with its constructive atmosphere was an important step towards better declarative programming languages.

Michael Hanus (RWTH Aachen)
John W. Lloyd (University of Bristol)
Juan José Moreno Navarro (Universidad Politécnica de Madrid)

Acknowledgements