Preface

This volume represents a first attempt to bring together ideas from two previously unrelated research areas, namely *Software Engineering* and *Computational Reflection* and to evaluate the benefits that each can bring to the other.

Computational reflection, or for short reflection, is a quite young discipline that is steadily attracting attention within the community of object-oriented researchers and practitioners. The properties of transparency, separation of concerns, and extensibility supported by reflection have largely been accepted as useful for software development and design. Reflective features have been included in successful software development technologies such as the Java™ language. Reflection has proved to be useful in some of the most challenging areas of software engineering, including component-based software development, as demonstrated by extensive use of the reflective concept of introspection in the Enterprise JavaBeans™ component technology. Nevertheless, there are still cognitive barriers separating reflection from the discipline of software engineering, and, more specifically, object-oriented reflection from object-oriented software engineering. Only a few authors have begun to explore the opportunities offered by the inter-disciplinary application of concepts from reflection and software engineering, that is, from the novel research area of reflective software engineering.

It is our belief that current trends in ongoing research in object-oriented reflection and software engineering clearly indicate that an inter-disciplinary approach would be of utmost relevance for both. The overall goal of this volume is to support the circulation of ideas between these disciplines. Several interactions can be expected to take place between software engineering and object-oriented reflection, some of which we cannot even foresee. Both the application of reflective techniques and concepts to software engineering and, vice versa, the application to object-oriented reflection of software engineering techniques, methodologies, and concepts, are likely to support improvement and deeper understanding of these areas.

Software engineering may benefit from a cross-fertilisation with object-oriented reflection in several ways. Reflective features such as transparency, separation of concerns, and extensibility are likely to be of increasing relevance in the modern software engineering scenario, where the trend is towards systems that exhibit sophisticated functional and non-functional requirements; that are built from independently developed and evolved COTS components; that support plug-and-play, end-user directed reconfigurability; that make extensive use of networking and internetworking; that can be automatically upgraded through the Internet; that are open; and so on. Several of these issues highlight the need for a system to manage itself to some extent, to inspect components' interfaces dynamically, to augment its application-specific functionality with additional properties, and so on. From a pragmatic point of view, several object-oriented reflection techniques and technologies lend themselves to be employed in ad-

dressing these issues. On a more conceptual level, several key object-oriented reflection principles could play an interesting role as general software design principles. Even more fundamentally, object-oriented reflection may provide a cleaner conceptual framework than that underlying the rather 'ad-hoc' solutions embedded in most commercial platforms and technologies, including component-based software development technologies, system management technologies, and so on. The transparent nature of reflection makes it well suited to address problems such as evolution of legacy systems, customizable software, product families, and more. The scope of application of object-oriented reflection concepts in software engineering conceptually spans activities related to all the phases of the software life-cycle, from analysis and architectural design to development, reuse, maintenance, and evolution.

The reverse also holds. In the last two decades, object-oriented reflection has generated a rich offspring in terms of reflective programming languages and reflective systems. The background of most researchers in the field is in disciplines that were traditionally insulated from software engineering (e.g., artificial intelligence). It is thus likely that several applications of software design and development concepts, principles, techniques, and methodologies from software engineering to object-oriented reflection are still to be clearly detected and investigated.

It should be highlighted that the purpose of supporting a dialogue among software engineering and object-oriented reflection researchers and practitioners is more than simply stimulating a (useful) cross-discipline application of established results. We believe that both disciplines have reached a point where each needs concepts from the other in order to achieve significant improvements in several areas.

During OOPSLA'99, the editors of this volume organised and held a workshop (OORaSE'99 - 1st OOPSLA Workshop on Reflection and Software Engineering) with the aim of focusing interest on this emerging area and providing a meeting-point for researchers working on ideas straddling the research topics. The event proved a success both for the interest aroused and for the quality of the contributions presented. This volume is a natural follow-up to the workshop. It contains the best contributions presented at the workshop, improved both by the exchange of opinions that the authors had with the others attendees, and by the advice given to them by the reviewers of the volume. The volume also contains some contributions from experts in this field of research.

We would like to thank all the researchers who submitted papers to our workshop, both for their interest in our proposal and for their efforts in the emerging area of reflection and software engineering. We would also like to thank the member of our programme committee:

Shigeru Chiba,   University of Tsukuba, Japan
Stéphane Ducasse, University of Geneve, Switzerland
Serge Demeyer,   University of Berne, Switzerland
                             University of Antwerp, Belgium
John Lamping,   Xerox Parc, USA
Satoshi Matsuoka, Tokyo Institute of Technology, Japan
Dave Thomas,   Founder OTI Inc. and President, Bedarra Corp., Canada

and the external reviewers:

Franz Achermann   University of Geneve, Switzerland
Massimo Ancona   University of Genova, Italy
Gregor Kiczales   Xerox Parc, USA
Cristina Videira Lopes Xerox Parc, USA

who helped us in judging and selecting the submitted works for presentation at the workshop and then helped to improve the best papers in order for them to be published in this book. We would also to thank the invited authors who accepted our invitation to contribute to this volume, and last but not least the Department of Informatics, System and Communication of the University of Milano Bicocca for its financial support.

2nd April 2000   W. Cazzola, R. J. Stroud, and F. Tisato
                                                                                      Editors