Series Editors
Gerhard Goos, Karlsruhe University, Germany
Juris Hartmanis, Cornell University, NY, USA
Jan van Leeuwen, Utrecht University, The Netherlands

Volume Editors
Jan Bosch
Department of Computer Science and Business Administration
University of Karlskrona/Ronneby
S-372 25 Ronneby, Sweden
E-mail: jan.bosch@ide.hk-r.se

Stuart Mitchell
Department of Computer Science, University of York
York, YO1 5DD, UK
E-mail: stuart@cs.york.ac.uk

Cataloging-in-Publication data applied for
Die Deutsche Bibliothek - CIP-Einheitsaufnahme

Object-oriented technology: ECOOP 97 workshop reader; proceedings / ECOOP 97 workshops, Jyväskylä, Finland, June 9 - 13, 1997. Jan Bosch ; Stuart Mitchell (ed.). - Berlin ; Heidelberg ; New York ; Barcelona ; Budapest ; Hong Kong ; London ; Milan ; Paris ; Santa Clara ; Singapore ; Tokyo : Springer, 1998 (Lecture notes in computer science ; Vol. 1357)
ISBN 3-540-64039-8

CR Subject Classification (1991): D.1-3, H.2

ISSN 0302-9743

This work is subject to copyright. All rights are reserved, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, re-use of illustrations, recitation, broadcasting, reproduction on microfilms or in any other way, and storage in data banks. Duplication of this publication or parts thereof is permitted only under the provisions of the German Copyright Law of September 9, 1965, in its current version, and permission for use must always be obtained from Springer-Verlag. Violations are liable for prosecution under the German Copyright Law.

© Springer-Verlag Berlin Heidelberg 1998
Printed in Germany

Typesetting: Camera-ready by author
SPIN 10661353 063142 – 5 4 3 2 1 0 Printed on acid-free paper
1 Introduction

Although becoming more accepted in software industry, object-oriented technology still is an active field of research with many issues remaining to be addressed. This workshop reader, in a way, presents the width of the ongoing research activities in object-orientation. However, we feel one can classify these activities into three categories:

- **Domain-specific**: Several activities focus on a single application, e.g. telecommunication, or computer-science, e.g. real-time and mobility, domain. Research tries to address the domain-specific problems of object-oriented technology.
- **Design issues**: Object-oriented design has been an issue for at least a decade, but one can identify an increasing focus on formal approaches and on the evolution and re-engineering of existing object-oriented software.
- **Beyond object-orientation**: The object-oriented paradigm will, at some point, be replaced by a subsequent paradigm and several research efforts investigate alternative or extended approaches. Examples are extended language expressiveness for, e.g. design patterns and frameworks, component-oriented programming and aspect-oriented programming.

2 Contents

The remainder of this book is a selection and re-iteration of the contributions to 12 workshops (of a total of 15) held during the ECOOP’97 conference. The workshops generally relate to one of the above categories.

Part I is concerned with the use of object-oriented technology for in telecommunication domain, in particular service engineering. Due to the breaking of the monopoly of traditional telecommunication providers, competition has increased rapidly. Since the competitive edge is in services rather than connections, the organisers predict a shift to a service-driven telecommunication market. To develop new services with short time-to-market, low complexity, high reusability, etc., the object-oriented paradigm is considered to be the most suitable candidate.
Reflective real-time object-oriented programming and systems are discussed in Part II. Real-time systems have traditionally been very static which results in changes to system behaviour during the software lifecycle being very costly. An important area of real-time research is how to make systems flexible and adaptive whilst retaining guarantees about their temporal properties. Reflection, i.e. the ability to perform computation on the system itself, is seen as a potential mechanism for achieving these goals.

Part III discussed the problems of conventional languages for the implementation of design patterns and object-oriented frameworks and new approaches to describing and implementing design patterns and frameworks. One can identify three categories of approaches, i.e. design environment support visualizing structures otherwise implicit, the generative approach, i.e. a tool generating a code skeleton that can be filled in with application-specific code, and programming language extensions that provide first-class representations for design patterns or framework concepts.

Part IV discusses the semantics of object-oriented modelling techniques. The participants, among others, reacted against the traditional signature-based way of describing standards: In addition to a precise syntax, also precise semantics needs to be specified. This would, among others, allow for detecting inconsistencies and inaccuracies and for comparing descriptions for their semantic equivalence.

Part V is concerned with distributed object-oriented computing, in particular methodological and implementation-oriented models and formal modelling techniques. The participants concluded that enabling technologies such as CORBA and Java RMI still lack a firm ground since no de-facto standard of distributed OO computing exists. On the other hand, it is unclear how formal notations such as Petri Nets, Estelle and Lotos can support the object-oriented paradigm. Finally, methodological support for these systems is still in the early stages.

Part VI studies the evolution and re-engineering of existing object-oriented software. Even though the object-oriented paradigm is often thought to be superior to older approaches with respect to maintenance, the large body of complex and expanding object-oriented software is placing an increasing burden on software engineers. The evolution and re-engineering of object-oriented software is a complex and multifaceted problem that needs to be studied from several perspectives.

The modelling of software processes and artifacts is discussed in Part VII. The relation to object-oriented technology is twofold. First, object-oriented software development requires explicit modelling of its processes and artifacts. Secondly, software processes and artifacts can be modelled using object-oriented principles.

Part VIII is concerned with component-oriented programming (COP). COP is often referred to as the next step beyond OOP and COP is, among others, different in the focus on constructing systems from independently developed components. Research issues include the relation to software architecture, the non-functional requirements that components should fulfil and the glue-ing of components during application construction.

Part IX presents the doctoral students workshop. Different from the other workshops that focus on a technical topic, this workshop presents a wide variety of research topics. Instead, the common denominator is the current occupation of its participants.
Nevertheless, the part presents an excellent overview of ongoing research activities in object-orientation.

The topic of Part X is object-oriented real-time systems. Despite early scepticism, object-oriented technology has become more widely used in real-time systems. However, there remain several research issues such as generating code from RT specifications, validation and testing of dynamic behaviour and the relation between object-oriented real-time software and conventional kernels. In this part the focus is on the modelling of object-oriented real-time system and on implementation techniques.

Aspect-oriented programming is discussed in Part XI. One can identify a general awareness that code tangling reduces the quality, e.g. maintainability and reusability, of software. AOP is concerned with approaches to separately describe the various aspects of a component or software system and to compose them at a later stage. However, various approaches to achieving this separation of concerns can be identified and the part defines a characterisation of AOP and identifies research issues.

Part XII is concerned with operating systems. Modern operating systems must, in addition to the traditional requirements of performance and usability, fulfil additional requirements such as flexibility, adaptability and scalability. The object-oriented paradigm provides properties that help achieve these requirements. Operating system issues discussed in this part are, among others, reflection, configuration of adaptable operating systems and virtual machines.

3 About the diversity of the submissions

A workshop reader is, by its nature, very diverse in its contributions. This reader is by no means an exception to that rule. The contributions have a wide diversity in topics as well as the form of presentation. Some workshops are represented only by a workshop report whereas other workshops primarily present the participants papers and spend limited effort on summarizing the workshop. As editors, we have given the organisers much freedom in the presentation of their workshop. We considered the contents of the parts more important than their uniformity and we do not believe that this would decrease readability.

4 Acknowledgements

An incredible amount of people has been involved in creating this workshop reader, in particular all authors and workshop organisers. As editors, we merely composed their contributions. This workshop reader presents an important contribution to object-oriented research and we would like to express our sincere appreciation to everyone involved.
Table of Contents

I: Object-Oriented Technology for Telecommunications Services Engineering
S. Znaty and J.-P. Hubaux

Telecommunications Services Engineering: Principles Architectures and Tools
S. Znaty and J.-P. Hubaux

Difficulties with Mapping OMT Specifications into GDMO
A. Meisingset

Towards a “Good” Functional and Executable Behavior Model
D. Sidou

The Formalization and Simulation of a Network Management Model
J. L. Tellez Portas, D. Watrin and T. Ben Meriem

OST – An Object-Oriented Computer Networks System Specification Tool
L. Granville, L. Gaspary and J. Almeida

Experiences in Telecommunications Protocols with an OO Based Implementation Framework
J. Harju, B. Silverajan and I. Toivanen

Rapid Network Management Application Development
L. Deri

Real-Time Object-Oriented Distributed Processing with COREMO
Z. Choukair and A. Beugnard

II: Reflective Real-Time Object-Oriented Programming and Systems
S. E. Mitchell and R. J. Stroud

Reflective ORBs: Supporting Robust, Time-Critical Distribution
A. Singhai, A. Sane and R. Campbell

Weak Protection for Reflective Operating Systems
S. Chiba, T. Nishimura, K. Kourai, A. Ohnoki and T. Masuda

Implementing Real-Time Actors with MetaJava
M. Golm and J. Kleinöder

Adaptive Scheduling Using Reflection
S. E. Mitchell, A. Burns and A. J. Wellings

Adaptive Fault-Tolerant Systems and Reflective Architectures
J. Goldberg and R. J. Stroud
III: Language Support for Design Patterns and Frameworks
J. Bosch, G. Hedin and K. Koskimies 89
Implementing GoF Design Patterns in BETA
E. Agerbo and A. Cornils 92
Message Passing Abstractions as Elementary Bricks for Design Pattern Implementation: An Experiment
S. Ducasse 96
Patterns of the Agenda
A. H. Eden and A. Yehudai 100
Design Patterns as Program Extracts
E. E. Jacobsen 105
Design Patterns vs. Language Design
J. Gil and D. H. Lorenz 108
Multi-level Patterns
F. Matthijs, W. Joosen, B. Robben, B. Vanhaute and P. Verbaeten 112
Architectural Abstractions for Frameworks
P. Nowack 116
A Framework for Processors of Visual Languages
A.-P. Tuovinen 119
Type Oriented Programming
K. De Volder and W. De Meuter 123
A Framework Registration Language
Y. Peres, J. W. Malcolm, P. Vortman and G. Zodik 126
Tool Support for Framework-Specific Language Extensions
E. Bjarnason and G. Hedin 129
Design Patterns & Frameworks: On the Issue of Language Support
J. Bosch 133
Language Support for Design Patterns Using Attribute Extension
G. Hedin 137
A Language Implementation Framework in Java
M. Harsu, J. Hautamäki and K. Koskimies 141

IV: Precise Semantics for Object-Oriented Modeling Techniques
H. Kilov and B. Rumpe 145
Ontology-Based Layered Semantics for Precise OA&D Modeling
J. Bézivin and R. Lemesle 151
Formalizing Object-Oriented Models in the Object Calculus
J. C. Bicarregui, K. C. Lano and T. S. E. Maibaum 155
<table>
<thead>
<tr>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set Theory as a Semantic Framework for Object-Oriented Modeling</td>
<td>161</td>
</tr>
<tr>
<td>B. Cohen</td>
<td></td>
</tr>
<tr>
<td>Unification of the Models for Types, Classes and State Machines</td>
<td>169</td>
</tr>
<tr>
<td>G. W. Daugherty</td>
<td></td>
</tr>
<tr>
<td>Integration of Z-based Semantics of OO-Notations</td>
<td>173</td>
</tr>
<tr>
<td>J. Ebert and R. Süttenbach</td>
<td></td>
</tr>
<tr>
<td>Integrated Informal Object-Oriented and Formal Modeling Techniques</td>
<td>178</td>
</tr>
<tr>
<td>R. B. France and J.-M. Bruel</td>
<td></td>
</tr>
<tr>
<td>Semantics Through Pictures</td>
<td>182</td>
</tr>
<tr>
<td>S. Kent, A. Hamie, J. Howse, F. Civello and R. Mitchell</td>
<td></td>
</tr>
<tr>
<td>Business Rules: From Business Specification to Design</td>
<td>188</td>
</tr>
<tr>
<td>H. Kilov and I. Simmonds</td>
<td></td>
</tr>
<tr>
<td>Liberating Object-Oriented Modeling from Programming-Level Abstractions</td>
<td>195</td>
</tr>
<tr>
<td>R. Kurki-Suonio and T. Mikkonen</td>
<td></td>
</tr>
<tr>
<td>Discovering, Displaying and Agreeing Semantics in an Environment of Conflicts</td>
<td>200</td>
</tr>
<tr>
<td>G. Mullery</td>
<td></td>
</tr>
<tr>
<td>Towards a Precise Semantics for Object-Oriented Modeling Techniques, R. Breu, R. Grosu, F. Huber, B. Rumpe and W. Schwerin</td>
<td>205</td>
</tr>
<tr>
<td>Formal Definition and Refinement of UML's Module/Package Concept</td>
<td>211</td>
</tr>
<tr>
<td>A. Schürr and A. J. Winter</td>
<td></td>
</tr>
<tr>
<td>Experience with Formal Specification of CMM and UML</td>
<td>216</td>
</tr>
<tr>
<td>J. Warmer, J. Hogg, S. Cook and B. Selic</td>
<td></td>
</tr>
<tr>
<td>V: Models, Formalisms and Methods for Object-Oriented</td>
<td>221</td>
</tr>
<tr>
<td>Distributed Computing</td>
<td></td>
</tr>
<tr>
<td>R. Bastide and D. Buchs</td>
<td></td>
</tr>
<tr>
<td>VI: FAMOOS Workshop on Object-Oriented Software</td>
<td>257</td>
</tr>
<tr>
<td>Evolution and Re-engineering</td>
<td></td>
</tr>
<tr>
<td>E. Casais, A. Jaasksi and T. Lindner</td>
<td></td>
</tr>
<tr>
<td>On Implementation and Extension Complexities</td>
<td>264</td>
</tr>
<tr>
<td>E. Kantorowitz</td>
<td></td>
</tr>
<tr>
<td>Analyzing Dependencies to Solve Low-Level Problems</td>
<td>266</td>
</tr>
<tr>
<td>T. Richner and R. Nebbe</td>
<td></td>
</tr>
<tr>
<td>Adding Architecture to Design</td>
<td>268</td>
</tr>
<tr>
<td>M. Ruonavaara</td>
<td></td>
</tr>
<tr>
<td>Analysis of Object-Oriented Programs Using Graphs</td>
<td>270</td>
</tr>
<tr>
<td>O. Ciupke</td>
<td></td>
</tr>
</tbody>
</table>
Extracting State Diagrams from Legacy Systems
T. Systa and K. Koskimies 272

Combining Software Descriptions
K. Mens, T. Mens, P. Steyaert and K. De Hondt 274

Re-engineering with UML
R. Trauter 276

Evolution of Telecommunication Software Using OO: Organizational and Technical Issues
H. Bachatène and P. Arnaud 278

Design Pattern Restructuring
S. Demeyer, T.-D. Meijler and M. Reiger 280

Similarity Measures for the Object Model
A. Sarirete and J. Vaucher 283

VII: Modeling Software Processes and Artifacts
K. van den Berg, M. Aksit, P. van den Broek, L. Osterwel, K. Lieberherr and F. Marcelloni 285

Revised PMLs and PSEEs for Industrial SPI
R. Conradi and C. Liu 289

Applying Fuzzy Logic Techniques in Object-Oriented Software Development
F. Marcelloni and M. Aksit 295

Towards a Dedicated Object-Oriented Software Process Modeling Language
W. Reimer, W. Schäfer and T. Schmal 299

The Object Model for a Product Based Development Process
P. Hruby 303

Active Software Artifacts
M. Aksit, F. Marcelloni, B. Tekinerdogan, K. van den Berg and P. van den Broek 307

General Software Process Organization in the OSMOSIS Project
J. Bézivin 311

Object-Oriented Modeling of Software Processes and Artifacts: Promises and Challenges
J. Han 315

Object-Oriented Model for Representing Software Production Processes
I. Bider and M. Khomyakov 319
VIII: Component-Oriented Programming
J. Bosch, C. Szyperski and W. Weck

Toward Automatic, Run-Time Fault Management for Component-Based Applications
V. Baggioolini and J. Harms

Formal Methods for Component Software: The Refinement Calculus Perspective
M. Büchi and E. Sekerinski

Reuse Contracts as Component Interface Descriptions
K. De Hondt, C. Lucas and P. Steyaert

A Control Model for the Dynamic Selection and Configuration of Software Components
P. Lalanda

Design and Evaluation of Distributed Component-Oriented Software Systems
M. Goedicke and T. Meyer

The Fragile Base Class Problem and Its Impact on Component Systems
L. Mikhajlov and E. Sekerinski

The Challenge of the Global Software Process
T. Murer

Coupling of Workflow and Component-Oriented Systems
S. Schreyjak

A Generator for Composition Interpreters
J. Steensgaard-Madsen

On the Addition of Properties to Components
J. M. Troya and A. Vallecillo

Adapting Object-Oriented Components
J. Bosch

Inheritance Using Contracts & Object Composition
W. Weck

IX: The 7th Workshop for PhD Students in Object-Oriented Systems
F. Gerhardt, L. Wohlrab and E. Ernst

X: Object-Oriented Real-Time Systems
L. Barroca, E. Durr and F. Terrier

Formalizing Real-Time System Design
K. C. Lano and S. Goldsack