

Commenced Publication in 1973

Founding and Former Series Editors:

Gerhard Goos, Juris Hartmanis, and Jan van Leeuwen

Editorial Board

David Hutchison

Lancaster University, UK

Takeo Kanade

Carnegie Mellon University, Pittsburgh, PA, USA

Josef Kittler

University of Surrey, Guildford, UK

Jon M. Kleinberg

Cornell University, Ithaca, NY, USA

Alfred Kobsa

University of California, Irvine, CA, USA

Friedemann Mattern

ETH Zurich, Switzerland

John C. Mitchell

Stanford University, CA, USA

Moni Naor

Weizmann Institute of Science, Rehovot, Israel

Oscar Nierstrasz

University of Bern, Switzerland

C. Pandu Rangan

Indian Institute of Technology, Madras, India

Bernhard Steffen

University of Dortmund, Germany

Madhu Sudan

Massachusetts Institute of Technology, MA, USA

Demetri Terzopoulos

University of California, Los Angeles, CA, USA

Doug Tygar

University of California, Berkeley, CA, USA

Gerhard Weikum

Max-Planck Institute of Computer Science, Saarbruecken, Germany

Bertrand Meyer Jim Woodcock (Eds.)

Verified Software: Theories, Tools, Experiments

First IFIP TC 2/WG 2.3 Conference, VSTTE 2005
Zurich, Switzerland, October 10-13, 2005
Revised Selected Papers and Discussions

Volume Editors

Bertrand Meyer

ETH Zurich, Department of Computer Science

Clausiusstr. 59, 8092 Zurich, Switzerland

E-mail: bertrand.meyer@inf.ethz.ch

Jim Woodcock

University of York, Department of Computer Science

Heslington, York YO10 5DD, UK

E-mail: jim@cs.york.ac.uk

Library of Congress Control Number: 2008930409

CR Subject Classification (1998): D.1.0, D.2, D.2.1, D.2.6, D.2.13, D.3.1, D.4.5, F.3.1, F.4.1, F.4.3

LNCS Sublibrary: SL 2 – Programming and Software Engineering

ISSN 0302-9743

ISBN-10 3-540-69147-2 Springer Berlin Heidelberg New York

ISBN-13 978-3-540-69147-1 Springer Berlin Heidelberg New York

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. Violations are liable to prosecution under the German Copyright Law.

Springer is a part of Springer Science+Business Media

springer.com

© IFIP International Federation for Information Processing 2008

Printed in Germany

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India

Printed on acid-free paper SPIN: 12282821 06/3180 5 4 3 2 1 0

Preface

A Step Towards Verified Software

Worries about the reliability of software are as old as software itself; techniques for allaying these worries predate even James King's 1969 thesis on "A program verifier." What gives the whole topic a new urgency is the conjunction of three phenomena: the blitz-like spread of software-rich systems to control ever more facets of our world and our lives; our growing impatience with deficiencies; and the development—proceeding more slowly, alas, than the other two trends—of techniques to ensure and verify software quality.

In 2002 Tony Hoare, one of the most distinguished contributors to these advances over the past four decades, came to the conclusion that piecemeal efforts are no longer sufficient and proposed a "Grand Challenge" intended to achieve, over 15 years, the production of a *verifying compiler*: a tool that while processing programs would also guarantee their adherence to specified properties of correctness, robustness, safety, security and other desirable properties. As Hoare sees it, this endeavor is not a mere research project, as might normally be carried out by one team or a small consortium of teams, but a momentous endeavor, comparable in its scope to the successful mission to send a man to the moon or to the sequencing of the human genome. It requires a cogent agenda with both scientific and engineering components; support from the scientific community and from governments; and the collaboration of many groups, spread across many countries and including many specialties, for the advancement of a common goal.

VSTTE — the first conference on Verified Software: Theories, Tools, Experiments —(<http://vstte.ethz.ch>) was the launching event for this undertaking. Held at ETH Zurich in October of 2005, the conference gathered about 100 participants, the elite of software verification from a dozen countries. The conference was by invitation only. Although of course not every top expert could be invited, and not all who were invited could come, the attendee list makes up a sample of the best work currently being pursued in software verification. The present volume is the record of this memorable event.

We have not just gathered the position papers contributed by most of the participants. Attendees of scientific conferences know well that the benefit does not just come from formal paper presentations; question-and-answer sessions after talks are another prime source of insights, especially with participants as distinguished as those of VSTTE. In an effort to increase the attraction of this book for readers who were not in Zurich, we recorded the discussions and include the transcripts here (with the exclusion of some sessions with no associated paper). To preserve the spontaneity of these exchanges, we have kept the editing to a minimum, essentially removing a few superfluous comments. (Because the process took some time, it also serves as our excuse for the delay between the conference and the present publication.) We hope that you will enjoy the added value of this material, and perhaps even that you will at times feel like you are actually attending a live VSTTE session, including the occasional inaudible comment.

From “Exclusive ors” to Conjunction

The field of software verification has long been marked by competition between approaches, often resulting in binary oppositions—view A vs. view B. The oppositions still exist, and help define a rough multi-dimensional classification of the techniques represented at VSTTE and elsewhere; in attempting this classification, however, we should note that the variants along each dimension are increasingly recognized as complementary rather than exclusive. Software verification is sufficiently hard that a Grand Challenge effort must glean help from every good idea.

The first distinction pits *management*-oriented approaches against those based on *technology*. VSTTE was technology-oriented and this volume emphasizes the latter kind, but you will find occasional references to project management, the political context, model-driven development, process models and the like. Bad management can be as effective as bad technology in destroying software quality. In the rest of this discussion we limit ourselves to technical aspects.

A thick line has long divided proponents of *constructive* approaches to software construction from people who understand “verification” in the same way Reagan did in his reaction to Gorbachev’s arms-reduction promises: “Trust, but *verify*.” This division can also be called “a priori vs. a posteriori”: build the software right, in the Dijkstra tradition of *A Discipline of Programming*, continued today by such techniques as proof-supported refinement, well represented in this volume; or scrutinize the result, with the help of powerful tools, for possible quality violations. This is an example of where opposition tends nowadays to yield to convergence: we need the best techniques to produce software that will have no bugs; and the best techniques to find any bugs that, all these efforts notwithstanding, still slip through the cracks.

Also prominent in this volume is the contrast between authors who concern themselves mostly with *specification*, often to the point of dismissing the advances of modern programming languages, and those who work at the implementation level, using these very programming languages as their basic tool. Even that conflict is not as irreducible as it sounds, since effective approaches using specification languages must go all the way to implementation, and advances in programming languages bring them to a level of abstraction not so far removed from the realm of specification.

The next opposition is between *static* approaches, which work on the basis of the software text only, and *dynamic* ones which require its execution. Examples of the first category are static analysis and proofs; the primary example of the second is testing. Although VSTTE did not prominently feature work on testing, this approach has made significant progress in recent years, and researchers are increasingly recognizing what software practitioners have known for years: that testing is today, and will remain for a long time, a key part of almost any effort to produce quality software.

Among static approaches, you will note the contrast between *full proofs* of correctness, which for any significant software system must command the support of powerful computer-based proof systems (the authors of several of the most famous of these tools were present at VSTTE and contributed to this volume), and *static analysis*, a term that generally describes ascertaining more partial properties of the software. Static analysis uses tools too, sometimes even the same theorem provers.

This is one more distinction that tends to get blurred, as the successes of partial static analysis continue to expand the scope and ambition of the properties they assess.

The distinction between static and dynamic approaches might seem to defy such blurring; but that too would be a wrong guess. The testing community's long-time differences with the advocates of proofs are fading out; a successor conference series of VSTTE, Tests And Proofs (TAP, <http://tap.ethz.ch>), launched at ETH Zurich in 2007, explores this new convergence. Also significant is the growing popularity of *model checking*, a technique that has enjoyed considerable successes in recent years, first with hardware verification and now with software. Model checking is similar to exhaustive testing in that it attempts to explore all execution paths of a program; but it does so on a simplified version of the program (to make the search tractable) and uses advanced proof techniques to limit the search space and make the results significant. One may further contrast model checking with *abstract interpretation*, also used successfully in large industrial projects; here too, some of the key contributors to both approaches were active participants in the conference and this book.

Competition turning into cooperation, “exclusive or” replaced by “and,” initial contradictions revealing complementarity the deeper one digs: this could be the theme of the present book. By exploring the authors' often contrasted contributions, you will discover both the multiplicity of viewpoints that exist today in the field of software verification and the numerous common themes and solutions that justify Tony Hoare's injunction: to join forces and all get to work together.

Acknowledgments. Many people made VSTTE and this volume possible. As Program Chair, Natarajan Shankar composed and orchestrated the event. We are of course particularly grateful to the authors, panelists and other participants who contributed their best work.

ETH Zurich (as part of its 150th anniversary celebrations) contributed a generous donation; so did Microsoft Research. We are particularly indebted to Professor Meinrad Eberle, coordinator of the ETH 150th anniversary, and Dr. Andrew Herbert, Managing Director of the Microsoft Research center in Cambridge.

ETH provided superb infrastructure and services, allowing in particular the recording of sessions as reflected in this book. The organizing team from the Chair of Software Engineering—Claudia Günthart, Bernd Schoeller, Ilinca Ciupa, Andreas Leitner, Manuel Oriol, Werner Dietl, Adam Darvas, Farhad Mehta, Luc de Louw and a number of student helpers—was essential to the preparation of the conference and its smooth operation, including the recordings. The grueling task of transcribing these recordings was performed with remarkable efficiency by Patrick Schönbach. Claudia Günthart helped put the book in its final form.

Our primary debt is to the two conference Chairs, Tony Hoare and Jay Misra, for not only conceiving the vision but also relentlessly pursuing its realization down to the most practical details, with an energy that never ceases to amaze us all. We hope that readers of this book will feel some of that energy, and share some of the excitement that pervaded every session of the VSTTE conference.

VSTTE Chairs and Committees

Joint General Chairs

Tony Hoare, Microsoft Research, UK
Jayadev Misra, University of Texas at Austin, USA

Program Chair

Natarajan Shankar, SRI, USA

Steering Committee

Manfred Broy, TU München, Germany	J Moore, University of Texas at Austin, USA
Butler Lampson, Microsoft Research, UK	Amir Pnueli, Weizmann Inst. of Science, Israel
Mathai Joseph, Tata Research Development and Design Centre, India	Michel Sintzoff, U. Cath. de Louvain, Belgium
Gilles Kahn, INRIA, France	Zhou Chao Chen, Chinese Acad. of Sciences

Program Committee

Jean-Raymond Abrial, ETH Zurich, Switzerland	Gerard Holzmann, Jet Propulsion Laboratory, USA
Dines Bjørner, Technical Univ. of Denmark	Cliff Jones, Univ. of Newcastle, UK
Patrick Cousot, École Normale Supérieure, France	Greg Morrisett, Harvard Univ., UK
Michael Ernst, Mass. Inst. of Technology, USA	George Necula, UC Berkeley, USA
David Evans, Univ. of Virginia, USA	Greg Nelson, HP Systems Research Center, USA
David Gries, Cornell University, USA	Tobias Nipkow, TU München, Germany
Orna Grumberg, Israel Inst. of Technology	Colin O'Halloran, QinetiQ, UK
Masami Hagiya, Univ. of Tokyo, Japan	Ernst-Rüdiger Olderog, Univ. Oldenburg, Germany
Ian Hayes, Univ. of Queensland, Australia	Mooly Sagiv, Tel-Aviv University, Israel

He Jifeng, United Nations Univ.,
Macao

Connie Heitmeyer, Naval Research
Laboratory, USA

Carolyn Talcott, SRI International, USA

Jian Zhang, Academia Sinica, China

Publication Chair

Jim Woodcock, University of York, UK

Publication Committee

David Gries, Cornell University, USA

Cliff Jones, Newcastle, UK

Bertrand Meyer, ETH Zurich, Switzerland

Ernst-Rüdiger Olderog, Univ. Oldenburg, Germany

Organization Chair

Bertrand Meyer, ETH Zurich, Switzerland

Organization Committee

Bernd Schoeller, ETH Zurich, Switzerland

Claudia Günthart, ETH Zurich, Switzerland



Tony Hoare, opening session



Rajeev Joshi, Joseph Kiniry, Greg Nelson, Natarajan Shankar, Jay Misra, Tony Hoare, closing panel



Welcome party in the historic Semper-Aula of the ETH



Panagiotis Manolios, Wolfgang Paul, Amir Pnueli



Cliff Jones, Connie Heitmeyer



Anthony Hall, Rod Chapman
in the “VSTTE Express” (chartered train to Üetliberg for conference dinner)



Jay Misra, Niklaus Wirth, Tony Hoare

Participants



Jean-Raymond Abrial, ETH Zurich, Switzerland

Rajeev Alur, University of Pennsylvania, USA

Myla Archer, Naval Research Laboratory, USA

Ralph Back, Abo Akademi University, Finland

Thomas Ball, Microsoft Research, USA

David Basin, ETH Zurich, Switzerland

Yves Bertot, INRIA, France

Ramesh Bharadwaj, Naval Research Laboratory, USA

Egon Börger, University of Pisa, Italy

Manfred Broy, TU Munich, Germany

Tevfik Bultan, University of California, USA

Michael Butler, University of Southampton, UK

Supratik Chakraborty, Indian Institute of Technology, India

Patrice Chalin, Concordia University, Canada

Roderick Chapman, Praxis High Integrity Systems, UK

Marsha Chechik, University of Toronto, Canada

Alessandro Coglio, Kestrel Institute, USA

Patrick Cousot, École Normale Supérieure, France

Michael Ernst, MIT, USA

Kathi Fisler, Worcester Polytechnic Institute, USA

Kokichi Futatsugi, JAIST, Japan
 Allen Goldberg, Kestrel Technology
 LLC, USA
 Cordell Green, Kestrel Institute, USA
 Arie Gurfinkel, University of Toronto,
 Canada
 Masami Hagiya, University of Tokyo,
 Japan
 Anthony Hall, Oxford, UK
 Stefan Hallerstede, ETH Zurich,
 Switzerland
 Klaus Havelund, Kestrel Technology
 LLC, USA
 Eric Hehner, University of Toronto,
 Canada
 Constance Heitmeyer, Naval Research
 Laboratory, USA
 Michael G. Hinchey, NASA, USA
 Thai Son Hoang, ETH Zurich,
 Switzerland
 Tony Hoare, Microsoft Research, UK
 Gerard J. Holzmann, NASA Jet
 Propulsion Laboratory, USA
 Peter Vincent Homeier, NSA, USA
 Andrew Ireland, Heriot-Watt University,
 UK
 Bart Jacobs, Radboud University
 Nijmegen, Netherlands
 He Jifeng, UNU-IIST, China
 Cliff Jones, Newcastle University, UK
 Mathai Joseph, Tata Consultancy
 Services, India
 Rajeev Joshi, NASA/JPL Laboratory for
 Reliable Software, USA
 Joseph Kiniry, University College
 Dublin, Ireland
 Daniel Kröning, ETH Zurich,
 Switzerland
 Patrick Lam, Massachusetts Institute of
 Technology, USA
 Gary T. Leavens, Iowa State University,
 USA
 Rustan Leino, Microsoft Research, USA
 Zhiming Liu, UNU-IIST, Macao, China
 Panagiotis Manolios, Georgia Tech,
 USA

Tiziana Margaria, University of
 Göttingen, Germany
 Edu Metz, Nokia Research Center, USA
 Bertrand Meyer, ETH Zurich, Switzerland
 Jayadev Misra, University of Texas at
 Austin, USA
 J. Strother Moore, University of Texas,
 USA
 Peter Müller, ETH Zurich, Switzerland
 David A. Naumann, Stevens Institute of
 Technology, USA
 Greg Nelson, USA
 Colin O'Halloran, QinetiQ, UK
 Peter O'Hearn, Queen Mary, University
 of London, UK
 Ernst-Rüdiger Olderog, University of
 Oldenburg, Germany
 Manuel Oriol, ETH Zurich, Switzerland
 Wolfgang Paul, University of
 Saarbrücken, Germany
 Amir Pnueli, Weizmann Institute, Israel,
 and New York University, USA
 Sanjiva Prasad, Indian Institute of
 Technology, India
 Ganesan Ramalingam, IBM Research,
 India
 Thomas Reps, University of Wisconsin-
 Madison, USA
 Tamara Rezk, INRIA, France
 Martin Rinard, Massachusetts Institute of
 Technology, USA
 Willem Paul de Roever, University of
 Kiel, Germany
 Grigore Rosu, University of Illinois, USA
 Harald Ruess, SRI International, USA
 John Rushby, SRI International, USA
 Shmuel Sagiv, Tel-Aviv University, Israel
 Peter Schmitt, University of Karlsruhe,
 Germany
 Florian Schneider, ETH Zurich,
 Switzerland
 Wolfram Schulte, Microsoft Research,
 USA
 Natarajan Shankar, SRI International,
 USA
 Natasha Sharygina, Carnegie Mellon
 University, USA

- | | |
|--|--|
| Michel Sintzoff, Université Catholique de Louvain, Belgium | Helmut Veith, TU Munich, Germany |
| Douglas R. Smith, Kestrel Institute, USA | Arnaud Venet, Kestrel Technology, USA |
| Graham Steel, University of Edinburgh, UK | Laurent Voisin, ETH Zurich, Switzerland |
| Bernhard Steffen, University Dortmund, Germany | Georg Weissbacher, ETH Zurich, Switzerland |
| Ofer Strichman, Technion, Israel | Niklaus Wirth, ETH Zurich, Switzerland |
| Aaron Stump, Washington University, USA | Jim Woodcock, University of York, UK |
| Carolyn Talcott, SRI International, USA | Stephan Zdancewic, University of Pennsylvania, USA |
| Cesare Tinelli, University of Iowa, USA | Naijun Zhan, Chinese Academy of Sciences |
| Mark Utting, University of Waikato, New Zealand | Jian Zhang, Chinese Academy of Sciences |
| Eric Van Wyk, University of Minnesota, USA | Jianjun Zhao, Fukuoka Institute of Technology, Japan |

Table of Contents

Introduction

Verified Software: Theories, Tools, Experiments: Vision of a Grand Challenge Project	1
<i>Tony Hoare and Jay Misra</i>	

Verification Tools

Towards a Worldwide Verification Technology	19
<i>Wolfgang Paul</i>	
It Is Time to Mechanize Programming Language Metatheory	26
<i>Benjamin C. Pierce, Peter Sewell, Stephanie Weirich, and Steve Zdancewic</i>	
Methods and Tools for Formal Software Engineering	31
<i>Zhiming Liu and R. Venkatesh</i>	

Guaranteeing Correctness

The Verified Software Challenge: A Call for a Holistic Approach to Reliability	42
<i>Thomas Ball</i>	
A Mini Challenge: Build a Verifiable Filesystem	49
<i>Rajeev Joshi and Gerard J. Holzmann</i>	
A Constructive Approach to Correctness, Exemplified by a Generator for Certified Java Card Applets	57
<i>Alessandro Coglio and Cordell Green</i>	
Some Interdisciplinary Observations about Getting the “Right” Specification	64
<i>Cliff B. Jones</i>	

Software Engineering Aspects

Software Verification and Software Engineering a Practitioner’s Perspective	70
<i>Anthony Hall</i>	
Decomposing Verification Around End-User Features	74
<i>Kathi Fisler and Shriram Krishnamurthi</i>	

Verifying Object-Oriented Programming

Automatic Verification of Strongly Dynamic Software Systems	82
<i>N. Dor, J. Field, D. Gopan, T. Lev-Ami, A. Loginov, R. Manevich, G. Ramalingam, T. Reps, N. Rinetzky, M. Sagiv, R. Wilhelm, E. Yahav, and G. Yorsh</i>	
Reasoning about Object Structures Using Ownership	93
<i>Peter Müller</i>	
Modular Reasoning in Object-Oriented Programming	105
<i>David A. Naumann</i>	
Scalable Specification and Reasoning: Challenges for Program Logic . . .	116
<i>Peter W. O'Hearn</i>	

Programming Language and Methodology Aspects

Lessons from the JML Project	134
<i>Gary T. Leavens and Curtis Clifton</i>	
The Spec# Programming System: Challenges and Directions	144
<i>Mike Barnett, Robert DeLine, Manuel Fähndrich, Bart Jacobs, K. Rustan M. Leino, Wolfram Schulte, and Herman Venter</i>	
Integrating Static Checking and Interactive Verification: Supporting Multiple Theories and Provers in Verification	153
<i>Joseph R. Kiniry, Patrice Chalin, and Clément Hurlin</i>	

Components

Automated Test Generation and Verified Software	161
<i>John Rushby</i>	
Dependent Types, Theorem Proving, and Applications for a Verifying Compiler	173
<i>Yves Bertot and Laurent Théry</i>	
Generating Programs Plus Proofs by Refinement	182
<i>Douglas R. Smith</i>	

Static Analysis

The Verification Grand Challenge and Abstract Interpretation	189
<i>Patrick Cousot</i>	
WYSINWYX: What You See Is Not What You eXecute	202
<i>G. Balakrishnan, T. Reps, D. Melski, and T. Teitelbaum</i>	

Implications of a Data Structure Consistency Checking System	214
<i>Viktor Kuncak, Patrick Lam, Karen Zee, and Martin Rinard</i>	
Towards the Integration of Symbolic and Numerical Static Analysis	227
<i>Arnaud Venet</i>	

Design, Analysis and Tools

Reliable Software Systems Design: Defect Prevention, Detection, and Containment	237
<i>Gerard J. Holzmann and Rajeev Joshi</i>	
Trends and Challenges in Algorithmic Software Verification	245
<i>Rajeev Alur</i>	
Model Checking: Back and Forth between Hardware and Software	251
<i>Edmund Clarke, Anubhav Gupta, Himanshu Jain, and Helmut Veith</i>	
Computational Logical Frameworks and Generic Program Analysis Technologies	256
<i>José Meseguer and Grigore Roşu</i>	

Formal Techniques

A Mechanized Program Verifier	268
<i>J. Strother Moore</i>	
Verifying Design with Proof Scores	277
<i>Kokichi Futatsugi, Joseph A. Goguen, and Kazuhiro Ogata</i>	
Integrating Theories and Techniques for Program Modelling, Design and Verification: Positioning the Research at UNU-IIST in Collaborative Research on the Verified Software Challenge	291
<i>Bernard K. Aichernig, He Jifeng, Zhiming Liu, and Mike Reed</i>	
Eiffel as a Framework for Verification	301
<i>Bertrand Meyer</i>	

Position Papers

Can We Build an Automatic Program Verifier? Invariant Proofs and Other Challenges	308
<i>Myla Archer</i>	
Verified Software: The <i>Real</i> Grand Challenge	318
<i>Ramesh Bharadwaj</i>	

Linking the Meaning of Programs to What the Compiler Can Verify	325
<i>Egon Börger</i>	
Scalable Software Model Checking Using Design for Verification	337
<i>Tevfik Bultan and Aysu Betin-Can</i>	
Model-Checking Software Using Precise Abstractions	347
<i>Marsha Chechik and Arie Gurfinkel</i>	
Toasters, Seat Belts, and Inferring Program Properties	354
<i>David Evans</i>	
On the Formal Development of Safety-Critical Software	362
<i>Andy Galloway, Frantz Iwu, John McDermid, and Ian Toyn</i>	
Verify Your Runs	374
<i>Klaus Havelund and Allen Goldberg</i>	
Specified Blocks	384
<i>Eric C.R. Hehner</i>	
A Case for Specification Validation	392
<i>Mats P.E. Heimdahl</i>	
Some Verification Issues at NASA Goddard Space Flight Center	403
<i>Michael G. Hinchey, James L. Rash, and Christopher A. Rouff</i>	
Performance Validation on Multicore Mobile Devices	413
<i>Thomas Hubbard, Raimondas Lencevicius, Edu Metz, and Gopal Raghavan</i>	
Tool Integration for Reasoned Programming	422
<i>Andrew Ireland</i>	
Decision Procedures for the Grand Challenge	428
<i>Daniel Kroening</i>	
The Challenge of Hardware-Software Co-verification	438
<i>Panagiotis Manolios</i>	
From the How to the What	448
<i>Tiziana Margaria and Bernhard Steffen</i>	
An Overview of Separation Logic	460
<i>John C. Reynolds</i>	
A Perspective on Program Verification	470
<i>Willem-Paul de Roever</i>	
Meta-Logical Frameworks and Formal Digital Libraries	478
<i>Carsten Schürmann</i>	

Languages, Ambiguity, and Verification	486
<i>The SPARK Team</i>	
The Importance of Non-theorems and Counterexamples in Program Verification	491
<i>Graham Steel</i>	
Regression Verification - A Practical Way to Verify Programs	496
<i>Ofer Strichman and Benny Godlin</i>	
Programming with Proofs: Language-Based Approaches to Totally Correct Software	502
<i>Aaron Stump</i>	
The Role of Model-Based Testing	510
<i>Mark Utting</i>	
Abstraction of Graph Transformation Systems by Temporal Logic and Its Verification	518
<i>Mitsuharu Yamamoto, Yoshinori Tanabe, Koichi Takahashi, and Masami Hagiya</i>	
Program Verification by Using DISCOVERER	528
<i>Lu Yang, Naijun Zhan, Bican Xia, and Chaochen Zhou</i>	
Constraint Solving and Symbolic Execution	539
<i>Jian Zhang</i>	
Author Index	545