Ivica Crnkovic
Judith A. Stafford
Heinz W. Schmidt
Kurt Wallnau (Eds.)

LNCS 3054

Component-Based Software Engineering

7th International Symposium, CBSE 2004
Edinburgh, UK, May 2004
Proceedings

Springer

Ivica Crnkovic Judith A. Stafford
Heinz W. Schmidt Kurt Wallnau (Eds.)

Component-Based Software Engineering

7th International Symposium, CBSE 2004
Edinburgh, UK, May 24-25, 2004
Proceedings

Springer

Volume Editors

Ivica Crnkovic
Department of Computer Science and Engineering
Mälardalen University
Box 883, 72123 Västerås, Sweden
E-mail: ivica.crnkovic@mdh.se

Judith A. Stafford
Department of Computer Science, Tufts University
161 College Avenue, Medford, MA 02155, USA
E-mail: jas@cs.tufts.edu

Heinz W. Schmidt
School of Computer Science and Software Engineering
Monash University
Wellington Road, Clayton VIC 3800 , Australia
E-mail: Heinz.Schmidt@csse.monash.edu.au

Kurt Wallnau
Software Engineering Institute, Carnegie Mellon University
Pittsburgh, Pennsylvania 15213-3890, USA
E-mail: kcw@sei.cmu.edu

Library of Congress Control Number: 2004105115

CR Subject Classification (1998): D.2, D.1.5, D.3, F.3.1

ISSN 0302-9743
ISBN 3-540-21998-6 Springer-Verlag Berlin Heidelberg New York

Springer-Verlag is a part of Springer Science+Business Media

springeronline.com

Printed in Germany

Typesetting: Camera-ready by author, data conversion by Olgun Computergrafik
Printed on acid-free paper SPIN: 11006947 06/3142 5 4 3 2 1 0

Lecture Notes in Computer Science 3054

Commenced Publication in 1973
Founding and Former Series Editors:
Gerhard Goos, Juris Hartmanis, and Jan van Leeuwen

Preface

Component-based software engineering (CBSE) is concerned with the development of software-intensive systems from reusable parts (components), the development of such reusable parts, and the maintenance and improvement of systems by means of component replacement and customization. Although it holds considerable promise, there are still many challenges facing both researchers and practitioners in establishing CBSE as an efficient and proven engineering discipline.

Six CBSE workshops have been held consecutively at the most recent six International Conferences on Software Engineering (ICSE). The premise of the last three CBSE workshops was that the long-term success of component-based development depends on the viability of an established science and technology foundation for achieving predictable quality in component-based systems.

The intent of the CBSE 2004 symposium was to build on this premise, and to provide a forum for more in-depth and substantive treatment of topics pertaining to predictability, to help establish cross-discipline insights, and to improve cooperation and mutual understanding. The goal of the CBSE 2004 symposium was to discuss and present more complete and mature works, and consequently collect the technical papers in published proceedings. The response to the Call for Papers was beyond expectations: 82 papers were submitted. Of those 25 (12 long and 13 short) were accepted for publication. In all 25 cases, the papers were reviewed by three to four independent reviewers. The symposium brought together researchers and practitioners from a variety of disciplines related to CBSE.

CBSE 2004 was privileged to have very competent, engaged and cooperative organizing and program committees with members involved in the forming of the symposium, its organization and in the review process. The review process, including the virtual review meetings, was organized completely electronically and succeeded thanks to the devoted work of the members and additional reviewers, and the excellent support from Richard van de Stadt who provided the electronic review system. The organizers of the ICSE 2004 conference, in particular Anthony Finkelstein, the General Chair, and Neno Medvidovic, the Workshops Chair, with great help and flexibility made it possible to organize CBSE 2004 as an adjunct event to the ICSE 2004 workshops. Springer-Verlag kindly agreed to publish the proceedings volume and helped greatly in its realisation. Finally all the contributors, the authors of the accepted papers, invited speakers and panelists contributed to the success of the symposium. We would like to thank each of them for their excellent contributions.

March 2004

Ivica Crnkovic
Heinz Schmidt
Judith Stafford
Kurt Wallanu

Message from the General Chair

Many hold that component software is the way to the next level of the software field's productivity. Others object that progress has been slow and that fundamental road blocks continue to be in the way. Ultimately, it is the need to move from manufacturing to an industrial approach that encourages the move away from monolithic software towards component-based engineering. Yet, it is true that much remains to be done and that component technologies available today have significant shortcomings. The same holds at the level of methodologies, processes, design and implementation languages, and tools.

The successful call for contributions to CBSE 2004 was a strong sign of the growing international attention. Research in academia and industry alike is embracing component software. With a maturing understanding of how components relate to other approaches, such as services and generators, the field is moving into a phase that promises good progress on both fundamental and practical issues. The broad range of topics covered by the authors of the accepted papers is a clear indication. From fundamental concerns of correctness and extrafunctional properties of composition to the architectural embedding of components, to methods and processes, and to the implications of using commercial off-the-shelf components – this symposium covers all of these topics.

With a strong and healthy community forming and growing, it was about time for CBSE to move from being a well-attended workshop to being a fully peer-reviewed and published symposium in its own right. This year's contributions inspire us to go that much further in the future. Hence, I am confident that we are seeing but the beginning of what I trust will develop into a successful series of events.

At this point, I would like to thank Ivica Crnkovic for running a smooth and efficient paper reviewing and selection process. Heinz Schmidt, Judy Stafford, and Kurt Wallnau supported the process greatly. I would also like to thank the two invited speakers, Hans Jonkers and Oscar Nierstrasz, who where quick to accept the invitation to speak at the newly shaped CBSE 2004 symposium, for delivering timely and thought-provoking contributions.

March 2004 Clemens Szyperski

Organization

CBSE 2004 was organized by Microsoft Research, USA, Monash University, Australia, Mälardalen University, Sweden, Carnegie Mellon University, USA and Tufts University, USA as an adjunct event to workshops at the 26th International Conference on Software Engineering (ICSE 2004).

Organizing Committee

General Chair	Clemens Szyperski (Microsoft Research, US)
Program Chair	Ivica Crnkovic (Mälardalen University, Sweden)
Program Co-chairs	Heinz Schmidt (Monash University, Australia)
	Judith Stafford (Tufts University, USA)
	Kurt Wallanu (SEI, Carnegie Mellon University, USA)

Program Committee

Uwe Aßmann, Linköping University, Sweden
Jakob Axelsson, Volvo Car Corporation, Sweden
Mike Barnett, Microsoft Research, USA
Judith Bishop, University of Pretoria, South Africa
Jan Bosch, University of Groningen, The Netherlands
Michel Chaudron, University of Eindhoven, The Netherlands
Wolfgang Emmerich, University College London, UK
Jacky Estublier, LSR-IMAG, France
Andre van der Hoek, University of California, Irvine, USA
Kathi Fisler, WPI, USA
Dimitra Giannakopoulou, NASA Ames, USA
Richard Hall, IMAG/LSR, France
Bengt Jonsson, Uppsala University, Sweden
Dick Hamlet, Portland State University, USA
George Heineman, WPI, USA
Paola Inverardi, University of L'Aquila, Italy
Shriram Krishnamurthi, Brown University, USA
Jeff Magee, Imperial College London, UK
Nenad Medvidovic, University of Southern California, USA
Magnus Larsson, ABB, Sweden
Rob van Ommering, Philips, The Netherlands
Heinz Schmidt, Monash University, Australia
Judith Stafford, Tufts University, USA
Dave Wile, Teknowledge, Corp., USA
Kurt Wallnau, SEI, Carnegie Mellon University, USA

Co-reviewers

Karine Arnout
Egor Bondarev
Ivor Bosloper
Reinder J. Bril
Marco Castaldi
Humberto Cervantes
Vittorio Cortellessa
Sybren Deelstra
Fernando Erazo
Howard Foster
Holger Hofmann
Anton Jansen
David N. Jansen
Michel Jaring
Eckhard Kruse
Christian Lange
Henrik Lonn
Chris Lueer
Sam Malek
Peter Mehlitz
Nikunj Mehta
Theo Dirk Meijler
Marija Mikic-Rakic
Raffaela Mirandola
Mohammad Reza Mousavi
Henry Muccini
Johan Muskens
Martin Naedele
Owen O'Malley
John Penix
Paul Pettersson
Noël Plouzeau
Manos Renieris
Roshanak Roshandel
Kevin B. Simons
Jorge Villalobos
Wang Yi

Previous events

CBSE 6 Workshop, Portland, USA (2003)
CBSE 5 Workshop, Orlando, USA (2002)
CBSE 4 Workshop, Toronto, Canada (2001)
CBSE 3 Workshop, Limerick, Ireland (2000)
CBSE 2 Workshop, Los Angeles, USA (1999)
CBSE 1 Workshop, Tokyo, Japan (1998)

Lecture Notes in Computer Science

For information about Vols. 1–2937

please contact your bookseller or Springer-Verlag

Vol. 3063: A. Llamosí, A. Strohmeier (Eds.), Reliable Software Technologies - Ada-Europe 2004. XIII, 333 pages. 2004.

Vol. 3060: A.Y. Tawfik, S.D. Goodwin (Eds.), Advances in Artificial Intelligence. XIII, 582 pages. 2004. (Subseries LNAI).

Vol. 3058: N. Sebe, M.S. Lew, T.S. Huang (Eds.), Computer Vision in Human-Computer Interaction. X, 233 pages. 2004.

Vol. 3054: I. Crnkovic, J.A. Stafford, H.W. Schmidt, K. Wallnau (Eds.), Component-Based Software Engineering. XI, 311 pages. 2004.

Vol. 3053: J. Davies, D. Fensel, C. Bussler, R. Studer (Eds.), The Semantic Web: Research and Applications. XIII, 490 pages. 2004.

Vol. 3042: N. Mitrou, K. Kontovasilis, G.N. Rouskas, I. Iliadis, L. Merakos (Eds.), NETWORKING 2004, Networking Technologies, Services, and Protocols; Performance of Computer and Communication Networks; Mobile and Wireless Communications. XXXIII, 1519 pages. 2004.

Vol. 3034: J. Favela, E. Menasalvas, E. Chávez (Eds.), Advances in Web Intelligence. XIII, 227 pages. 2004. (Subseries LNAI).

Vol. 3033: M. Li, X.-H. Sun, Q. Deng, J. Ni (Eds.), Grid and Cooperative Computing. XXXVIII, 1076 pages. 2004.

Vol. 3032: M. Li, X.-H. Sun, Q. Deng, J. Ni (Eds.), Grid and Cooperative Computing. XXXVII, 1112 pages. 2004.

Vol. 3031: A. Butz, A. Krüger, P. Olivier (Eds.), Smart Graphics. X, 165 pages. 2004.

Vol. 3028: D. Neuenschwander, Probabilistic and Statistical Methods in Cryptology. X, 158 pages. 2004.

Vol. 3027: C. Cachin, J. Camenisch (Eds.), Advances in Cryptology - EUROCRYPT 2004. XI, 628 pages. 2004.

Vol. 3026: C. Ramamoorthy, R. Lee, K.W. Lee (Eds.), Software Engineering Research and Applications. XV, 377 pages. 2004.

Vol. 3025: G.A. Vouros, T. Panayiotopoulos (Eds.), Methods and Applications of Artificial Intelligence. XV, 546 pages. 2004. (Subseries LNAI).

Vol. 3024: T. Pajdla, J. Matas (Eds.), Computer Vision - ECCV 2004. XXVIII, 621 pages. 2004.

Vol. 3023: T. Pajdla, J. Matas (Eds.), Computer Vision - ECCV 2004. XXVIII, 611 pages. 2004.

Vol. 3022: T. Pajdla, J. Matas (Eds.), Computer Vision - ECCV 2004. XXVIII, 621 pages. 2004.

Vol. 3021: T. Pajdla, J. Matas (Eds.), Computer Vision - ECCV 2004. XXVIII, 633 pages. 2004.

Vol. 3019: R. Wyrzykowski, J. Dongarra, M. Paprzycki, J. Wasniewski (Eds.), Parallel Processing and Applied Mathematics. XIX, 1174 pages. 2004.

Vol. 3015: C. Barakat, I. Pratt (Eds.), Passive and Active Network Measurement. XI, 300 pages. 2004.

Vol. 3012: K. Kurumatani, S.-H. Chen, A. Ohuchi (Eds.), Multi-Agnets for Mass User Support. X, 217 pages. 2004. (Subseries LNAI).

Vol. 3011: J.-C. Régin, M. Rueher (Eds.), Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems. XI, 415 pages. 2004.

Vol. 3010: K.R. Apt, F. Fages, F. Rossi, P. Szeredi, J. Váncza (Eds.), Recent Advances in Constraints. VIII, 285 pages. 2004. (Subseries LNAI).

Vol. 3009: F. Bomarius, H. Iida (Eds.), Product Focused Software Process Improvement. XIV, 584 pages. 2004.

Vol. 3008: S. Heuel, Uncertain Projective Geometry. XVII, 205 pages. 2004.

Vol. 3007: J.X. Yu, X. Lin, H. Lu, Y. Zhang (Eds.), Advanced Web Technologies and Applications. XXII, 936 pages. 2004.

Vol. 3006: M. Matsui, R. Zuccherato (Eds.), Selected Areas in Cryptography. XI, 361 pages. 2004.

Vol. 3005: G.R. Raidl, S. Cagnoni, J. Branke, D.W. Corne, R. Drechsler, Y. Jin, C.G. Johnson, P. Machado, E. Marchiori, F. Rothlauf, G.D. Smith, G. Squillero (Eds.), Applications of Evolutionary Computing. XVII, 562 pages. 2004.

Vol. 3004: J. Gottlieb, G.R. Raidl (Eds.), Evolutionary Computation in Combinatorial Optimization. X, 241 pages. 2004.

Vol. 3003: M. Keijzer, U.-M. O'Reilly, S.M. Lucas, E. Costa, T. Soule (Eds.), Genetic Programming. XI, 410 pages. 2004.

Vol. 3002: D.L. Hicks (Ed.), Metainformatics. X, 213 pages. 2004.

Vol. 3001: A. Ferscha, F. Mattern (Eds.), Pervasive Computing. XVII, 358 pages. 2004.

Vol. 2999: E.A. Boiten, J. Derrick, G. Smith (Eds.), Integrated Formal Methods. XI, 541 pages. 2004.

Vol. 2998: Y. Kameyama, P.J. Stuckey (Eds.), Functional and Logic Programming. X, 307 pages. 2004.

Vol. 2997: S. McDonald, J. Tait (Eds.), Advances in Information Retrieval. XIII, 427 pages. 2004.

Vol. 2996: V. Diekert, M. Habib (Eds.), STACS 2004. XVI, 658 pages. 2004.

Vol. 2995: C. Jensen, S. Poslad, T. Dimitrakos (Eds.), Trust Management. XIII, 377 pages. 2004.

Vol. 2994: E. Rahm (Ed.), Data Integration in the Life Sciences. X, 221 pages. 2004. (Subseries LNBI).

Vol. 2993: R. Alur, G.J. Pappas (Eds.), Hybrid Systems: Computation and Control. XII, 674 pages. 2004.

Vol. 2992: E. Bertino, S. Christodoulakis, D. Plexousakis, V. Christophides, M. Koubarakis, K. Böhm, E. Ferrari (Eds.), Advances in Database Technology - EDBT 2004. XVIII, 877 pages. 2004.

Vol. 2991: R. Alt, A. Frommer, R.B. Kearfott, W. Luther (Eds.), Numerical Software with Result Verification. X, 315 pages. 2004.

Vol. 2989: S. Graf, L. Mounier (Eds.), Model Checking Software. X, 309 pages. 2004.

Vol. 2988: K. Jensen, A. Podelski (Eds.), Tools and Algorithms for the Construction and Analysis of Systems. XIV, 608 pages. 2004.

Vol. 2987: I. Walukiewicz (Ed.), Foundations of Software Science and Computation Structures. XIII, 529 pages. 2004.

Vol. 2986: D. Schmidt (Ed.), Programming Languages and Systems. XII, 417 pages. 2004.

Vol. 2985: E. Duesterwald (Ed.), Compiler Construction. X, 313 pages. 2004.

Vol. 2984: M. Wermelinger, T. Margaria-Steffen (Eds.), Fundamental Approaches to Software Engineering. XII, 389 pages. 2004.

Vol. 2983: S. Istrail, M.S. Waterman, A. Clark (Eds.), Computational Methods for SNPs and Haplotype Inference. IX, 153 pages. 2004. (Subseries LNBI).

Vol. 2982: N. Wakamiya, M. Solarski, J. Sterbenz (Eds.), Active Networks. XI, 308 pages. 2004.

Vol. 2981: C. Müller-Schloer, T. Ungerer, B. Bauer (Eds.), Organic and Pervasive Computing – ARCS 2004. XI, 339 pages. 2004.

Vol. 2980: A. Blackwell, K. Marriott, A. Shimojima (Eds.), Diagrammatic Representation and Inference. XV, 448 pages. 2004. (Subseries LNAI).

Vol. 2979: I. Stoica, Stateless Core: A Scalable Approach for Quality of Service in the Internet. XVI, 219 pages. 2004.

Vol. 2978: R. Groz, R.M. Hierons (Eds.), Testing of Communicating Systems. XII, 225 pages. 2004.

Vol. 2977: G. Di Marzo Serugendo, A. Karageorgos, O.F. Rana, F. Zambonelli (Eds.), Engineering Self-Organising Systems. X, 299 pages. 2004. (Subseries LNAI).

Vol. 2976: M. Farach-Colton (Ed.), LATIN 2004: Theoretical Informatics. XV, 626 pages. 2004.

Vol. 2973: Y. Lee, J. Li, K.-Y. Whang, D. Lee (Eds.), Database Systems for Advanced Applications. XXIV, 925 pages. 2004.

Vol. 2972: R. Monroy, G. Arroyo-Figueroa, L.E. Sucar, H. Sossa (Eds.), MICAI 2004: Advances in Artificial Intelligence. XVII, 923 pages. 2004. (Subseries LNAI).

Vol. 2971: J.I. Lim, D.H. Lee (Eds.), Information Security and Cryptology -ICISC 2003. XI, 458 pages. 2004.

Vol. 2970: F. Fernández Rivera, M. Bubak, A. Gómez Tato, R. Doallo (Eds.), Grid Computing. XI, 328 pages. 2004.

Vol. 2968: J. Chen, S. Hong (Eds.), Real-Time and Embedded Computing Systems and Applications. XIV, 620 pages. 2004.

Vol. 2967: S. Melnik, Generic Model Management. XX, 238 pages. 2004.

Vol. 2966: F.B. Sachse, Computational Cardiology. XVIII, 322 pages. 2004.

Vol. 2965: M.C. Calzarossa, E. Gelenbe, Performance Tools and Applications to Networked Systems. VIII, 385 pages. 2004.

Vol. 2964: T. Okamoto (Ed.), Topics in Cryptology – CT-RSA 2004. XI, 387 pages. 2004.

Vol. 2963: R. Sharp, Higher Level Hardware Synthesis. XVI, 195 pages. 2004.

Vol. 2962: S. Bistarelli, Semirings for Soft Constraint Solving and Programming. XII, 279 pages. 2004.

Vol. 2961: P. Eklund (Ed.), Concept Lattices. IX, 411 pages. 2004. (Subseries LNAI).

Vol. 2960: P.D. Mosses (Ed.), CASL Reference Manual. XVII, 528 pages. 2004.

Vol. 2959: R. Kazman, D. Port (Eds.), COTS-Based Software Systems. XIV, 219 pages. 2004.

Vol. 2958: L. Rauchwerger (Ed.), Languages and Compilers for Parallel Computing. XI, 556 pages. 2004.

Vol. 2957: P. Langendoerfer, M. Liu, I. Matta, V. Tsaoussidis (Eds.), Wired/Wireless Internet Communications. XI, 307 pages. 2004.

Vol. 2956: A. Dengel, M. Junker, A. Weisbecker (Eds.), Reading and Learning. XII, 355 pages. 2004.

Vol. 2954: F. Crestani, M. Dunlop, S. Mizzaro (Eds.), Mobile and Ubiquitous Information Access. X, 299 pages. 2004.

Vol. 2953: K. Konrad, Model Generation for Natural Language Interpretation and Analysis. XIII, 166 pages. 2004. (Subseries LNAI).

Vol. 2952: N. Guelfi, E. Astesiano, G. Reggio (Eds.), Scientific Engineering of Distributed Java Applications. X, 157 pages. 2004.

Vol. 2951: M. Naor (Ed.), Theory of Cryptography. XI, 523 pages. 2004.

Vol. 2949: R. De Nicola, G. Ferrari, G. Meredith (Eds.), Coordination Models and Languages. X, 323 pages. 2004.

Vol. 2948: G.L. Mullen, A. Poli, H. Stichtenoth (Eds.), Finite Fields and Applications. VIII, 263 pages. 2004.

Vol. 2947: F. Bao, R. Deng, J. Zhou (Eds.), Public Key Cryptography – PKC 2004. XI, 455 pages. 2004.

Vol. 2946: R. Focardi, R. Gorrieri (Eds.), Foundations of Security Analysis and Design II. VII, 267 pages. 2004.

Vol. 2943: J. Chen, J. Reif (Eds.), DNA Computing. X, 225 pages. 2004.

Vol. 2941: M. Wirsing, A. Knapp, S. Balsamo (Eds.), Radical Innovations of Software and Systems Engineering in the Future. X, 359 pages. 2004.

Vol. 2940: C. Lucena, A. Garcia, A. Romanovsky, J. Castro, P.S. Alencar (Eds.), Software Engineering for Multi-Agent Systems II. XII, 279 pages. 2004.

Vol. 2939: T. Kalker, I.J. Cox, Y.M. Ro (Eds.), Digital Watermarking. XII, 602 pages. 2004.

Table of Contents

Invited Talks

Generation and Adaptation of Component-Based Systems

Tools and Building Framework

Components for Real-Time Embedded Systems

Extra-Funtional Properties of Components and Component-Based Systems

Measurements and Prediction Models for Component Assemblies

Putting Change at the Center of the Software Process*

Oscar Nierstrasz

Software Composition Group, University of Bern
www.iam.unibe.ch/~scg

Introduction

For over thirty years now, software components have been perceived as being essential stepping stones towards flexible and maintainable software systems. But where do the components come from? Once we have the components, how do we put them together? And when we are missing components, how should we synthesize them?

Lehman and Belady established in a classic study that a number of "Laws" of Software Evolution apply to successful software projects [10]. Of these, the two most insightful are perhaps:

- *Continuing change:* A program that is used in a real-world environment *must change*, or become progressively less useful in that environment.
- *Increasing complexity:* As a program evolves, it becomes *more complex*, and extra resources are needed to preserve and simplify its structure.

In this light we can observe that many recent trends in software engineering can actually be seen as *obstacles* to progress, since they offer metaphors that do not help address these issues [11]. "Software Engineering" itself can be seen as a dangerous metaphor that draws too strong an analogy between engineering of hardware and software. Similarly "software maintenance" is clearly a lie when we consider that real maintenance tasks are actually continuous development.

We know that successful software systems are doomed to change. But our programming languages and tools continue to focus on developing static, unchanging models of software. We propose that change should be at the *center* of our software process. To that end, we are exploring programming language mechanisms to support both *fine-grained composition* and *coarse-grained extensibility*, and we are developing tools and techniques to *analyse and facilitate change* in complex systems. In this talk we review problems and limitations with object-oriented and component-based development approaches, and we explore both technological and methodological ways in which change can be better accommodated.

* Extended summary of an invited talk at CBSE 2004 — International Symposium on Component-Based Software Engineering — Edinburgh, Scotland, May 24-25, 2004.

I. Crnkovic et al. (Eds.): CBSE 2004, LNCS 3054, pp. 1–4, 2004.

Language Support for Composition

What programming languages provide specific mechanisms that either take into account or support the fact that programs change over time? It is notable that mainstream programming languages largely emphasize the construction of static software structures, and disregard the fact that these structures are likely to change. We have been experimenting with various programming languages and language extensions that address certain aspects of change.

Piccola is a small language for composing applications from software components [1, 13]. Whereas we have many programming languages that are well-suited for building components, few focus on how components are put together. Piccola provides a notion of *first-class namespaces* that turns out to be immensely useful for expressing, packaging and controlling the ways in which software components are composed [12].

Traits are a fine-grained mechanism for decomposing classes into sets of related methods [15]. Traits overcome a number of difficulties with single and multiple inheritance, while avoiding the fragility inherent in mixins by sidestepping traditional linearization algorithms for composing features. Traits have proven to be extremely useful in refactoring complex class libaries [6].

Classboxes offer a minimal module system for controlling class extensions [4]. Class extensions support *unanticipated change* to third-party classes where subclassing is not an option. In classboxes, as in traits and Piccola, we note that the notion of first-class namespaces is an important means to manage and control change. We conjecture that programming languages that better support change will place more emphasis on such mechanisms.

Mining Components

Support for change is clearly not just a language issue. We also need good *tools* to analyze and manage code.

We have been developing a reengineering platform called *Moose* that serves as a code repository and a basis for analyzing software systems [7]. In this context we have developed a series of tools to aid in the understanding and restructuring of complex software systems.

CodeCrawler is a software visualization tool based on the notion of *polymetric views* — simple graphical visualizations of of direct software metrics [8]. One of the most striking applications of polymetrics views is in analyzing the evolution of a software system [9]: an *evolution matrix* quickly reveals which parts of a system are stable or undergoing change. We are further exploring the use of historical data to *predict change* in software systems [14].

We are also exploring ways to mine recurring structures from software systems. *ConAn* is a tool that applies *formal concept analysis* to detect recurring "concepts" in models of software. We have applied this approach to detect implicit contracts in class hierarchies [3] and to detect recurring "software patterns" [2]. We are now exploring ways to assess and improve the quality of the module structure of applications with respect to various reengineering operations.

Where Are We? Where Do We Go?

To conclude, we would do well to note that change is inevitable in software. As a consequence software components, being the *stable* part of software systems, can offer at most half of any equation that would help to improve software productivity.

There is a need for both languages and tools that offer better support to help us cope with and even exploit change.

Nevertheless, we should beware that any new techniques or methods carry some danger with them. Not only do metaphors sometimes blind us, but, as Berry points out [5], any technique that addresses a key difficulty in software development typically entails some painful steps that we will seek to avoid. To achieve any benefit, we must first overcome this pain.

Acknowledgments

We gratefully acknowledge the financial support of the Swiss National Science Foundation for the project "Tools and Techniques for Decomposing and Composing Software" (SNF Project No. 2000-067855.02, Oct. 2002 - Sept. 2004).

References

1. Franz Achermann, Markus Lumpe, Jean-Guy Schneider, and Oscar Nierstrasz. Piccola — a small composition language. In Howard Bowman and John Derrick, editors, *Formal Methods for Distributed Processing — A Survey of Object-Oriented Approaches*, pages 403–426. Cambridge University Press, 2001.
2. Gabriela Arévalo, Frank Buchli, and Oscar Nierstrasz. Software pattern detection using formal concept analysis. Submitted for publication, March 2004.
3. Gabriela Arévalo, Stéphane Ducasse, and Oscar Nierstrasz. X-Ray views: Understanding the internals of classes. In *Proceedings of ASE 2003*, pages 267–270. IEEE Computer Society, October 2003. short paper.
4. Alexandre Bergel, Stéphane Ducasse, and Roel Wuyts. Classboxes: A minimal module model supporting local rebinding. In *Proceedings of JMLC 2003 (Joint Modular Languages Conference)*, volume 2789 of *LNCS*, pages 122–131. Springer-Verlag, 2003. Best award paper.
5. Daniel Berry. The inevitable pain of software development: Why there is no silver bullet. In *Proceedings Radical Innovations of Software and Systems Engineering in the Future*, Venice, Italy, October 2002. preprint.
6. Andrew P. Black, Nathanael Schärli, and Stéphane Ducasse. Applying traits to the Smalltalk collection hierarchy. In *Proceedings OOPSLA '03, ACM SIGPLAN Notices*, volume 38, pages 47–64, October 2003.
7. Stéphane Ducasse, Michele Lanza, and Sander Tichelaar. Moose: an extensible language-independent environment for reengineering object-oriented systems. In *Proceedings of the Second International Symposium on Constructing Software Engineering Tools (CoSET 2000)*, June 2000.

8. Michele Lanza. Program visualization support for highly iterative development environments. In *Proceedings of VisSoft 2003 (International Workshop on Visualizing Software for Understanding and Analysis)*, page To appear. IEEE Press, 2003.
9. Michele Lanza and Stéphane Ducasse. Understanding software evolution using a combination of software visualization and software metrics. In *Proceedings of LMO 2002 (Langages et Modèles à Objets*, pages 135–149, 2002.
10. Manny M. Lehman and Les Belady. *Program Evolution — Processes of Software Change*. London Academic Press, 1985.
11. Oscar Nierstrasz. Software evolution as the key to productivity. In *Proceedings Radical Innovations of Software and Systems Engineering in the Future*, Venice, Italy, October 2002. preprint.
12. Oscar Nierstrasz and Franz Achermann. Separating concerns with first-class namespaces. In Tzilla Elrad, Siobán Clarke, Mehmet Aksit, and Robert Filman, editors, *Aspect-Oriented Software Development*. Addison-Wesley, 2004. To appear.
13. Oscar Nierstrasz, Franz Achermann, and Stefan Kneubühl. A guide to JPiccola. Technical Report IAM-03-003, Institut für Informatik, Universität Bern, Switzerland, June 2003.
14. Daniel Ratiu, Stéphane Ducasse, Tudor Gîrba, and Radu Marinescu. Using history information to improve design flaws detection. In *Proceedings of the Conference on Software Maintenance and Reengineering (CSMR 2004)*, 2004.
15. Nathanael Schärli, Stéphane Ducasse, Oscar Nierstrasz, and Andrew Black. Traits: Composable units of behavior. In *Proceedings ECOOP 2003*, volume 2743 of *LNCS*, pages 248–274. Springer Verlag, July 2003.