ISSN 1000-9825, CODEN RUXUEW
Journal of Software, Vol.17, No.4, April 2006, pp.649−657
DOI: 10.1360/jos170649

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel/Fax: +86-10-62562563

Unifying the Software Process Spectrum*

Mingshu Li[1+], Barry W. Boehm[2], Leon J. Osterweil[3]

[1](State Key Lab. of Computer Science and Lab. for Internet Software Technologies, Institute of Software at Chinese Academy of Sciences, Beijing 100080, China)

[2](University of Southern California, CA 90089-0781, USA)

[3](University of Massachusetts Amherst, Amherst MA 01003, USA)

+ Corresponding author: Phn: +86-10-62635241, E-mail: mingshu@iscas.ac.cn, http://www.iscas.ac.cn

Li MS, Boehm BW, Osterweil LJ. Unifying the software process spectrum. *Journal of Software*, 2006,17(4): 649−657. http://www.jos.org.cn/1000-9825/17/649.htm

Abstract: Software Process Workshop (SPW 2005) was held in Beijing on May 25-27, 2005. This paper introduces the motivation of organizing such a workshop, as well as its theme and paper gathering and review; and summarizes the main content and insights of 11 keynote speeches, 30 regular papers in five sessions of “Process Content”, “Process Tools and Metrics”, “Process Management”, “Process Representation and Analysis”, and “Experience Reports”, 8 software development support tools demonstration, and the ending panel “Where Are We Now? Where Should We Go Next?”.

Key words: software process; microprocess and macroprocess; workshop

A software process can be defined as the coherent set of policies, organizational structures, technologies, procedures, and artifacts that are needed to conceive, develop, deploy, and maintain a software product[1]. Each process can be described in a variety of ways, using text, pictures, or a combination. Software engineering researchers have suggested a variety of formats for such descriptions, usually organized as a model that contains key processes features[2].

Boehm[3] viewed the software development process in light of the risks involved, suggesting that a spiral model could combine development activities with risk management to minimize and control risk. The major distinguishing feature of the spiral model is that it creates a risk-driven approach to the software process rather than a primarily document-driven or code-driven process. It incorporates many of the strengths of other models and resolves many of their difficulties.

Osterweil[4] proposed that the notion of a “process program”—namely an object which has been created by a development process, and which is itself a software process description—should become a key focus of software engineering research and practice. His paper suggesting the idea of process programming was awarded as a 10-year retrospective and could be considered as a pivotal paper in modern research on software processes.

The above only lists two of the most influential papers published in the mid-1980s. It is a “golden time” for software process research between mid-1980s and mid-1990s. The International Software Process Association (ISPA) sponsors a series of International Software Process Workshops (ISPW), the first one was started in 1984 at Egham, Sueery, UK; then followed ISPW 2 (software process and software environments) in 1985, ISPW 3

* Received 2005-12-20; Accepted 2006-03-06

(iteration in the software process) in 1986, ISPW 4 (representing and enacting the software process) in 1988, ISPW 5 (experience with software process models) in 1989, ISPW 6 (support for the software process) in 1990, ISPW 7 (communication and coordination in the software process) in 1991, ISPW 8 (state of the practice in process technology) in 1993, ISPW 9 (the role of humans in the process) in 1994 and ISPW 10 (process support of software product Lines) in 1996.

Scheduled on June 1998, the theme of ISPW 11 was "Expanding Our Horizons: Building Bridges Beyond Software Process". However, the Program Chair of ISPW11, Gail E. Kaiser, on 28 March 1998, announced that, "ISPW11 will NOT be held in June as previously announced, but will be postponed indefinitely…". Another series sponsored by ISPA, the International Conference on Software Process (ICSP) starting from 1991, was also interrupted in 1998 at ICSP 5, which attempted to leverage that overlap by bringing together the researchers and practitioners from these different communities to explicitly identify how the individual technologies they are producing can be combined to support organized teams.

Though having some other related events, it is almost 10 years not bring together international leaders in the software process research area. A watershed event for the software process community is needed. That is the 2005 Beijing Software Process Workshop (SPW 2005). It was held in Beijing, P. R. China on May 25-27, 2005.

1 The Theme of SPW2005

The theme of SPW2005 is: Unifying the Software Process Spectrum.

Software Process encompasses all the activities that aim at developing or evolving software products. The expanding role of software and information systems in the world has focused increasing attention upon the need for assurances that software systems can be developed at acceptable speed and cost, on a predictable schedule, and in such a way that resulting systems are of acceptably high quality and can be evolved surely and rapidly as usage contexts change. This sharpened focus is creating new challenges and opportunities for software process technology. The increasing pace of software system change requires more lightweight and adaptive processes, while the increasing mission-criticality of software systems requires more process predictability and control, as well as more explicit attention to business or mission values. Emergent application requirements create a need for ambiguity-tolerance. Systems of systems and global development create needs for scalability and multi-collaborator, multi-culture concurrent coordination. COTS products provide powerful capabilities, but their vendor-determined evolution places significant constraints on software definition, development, and evolution processes.

The recognition of these needs has spawned a considerable amount of software process research across a broad spectrum. Much of the research has addressed the overall characteristics and needs of software processes, focusing on such issues as process architectures, process behavioral characteristics, and how processes fit with higher level organizational systems and characteristics. We refer to these investigations as macroprocess research. Simultaneously there has also been considerable research directed towards the precise, complete, detailed and unambiguous definition of software processes, focusing on such issues as detection of process flaws, and facilitation of the human-machine synergies inherent in software processes. We refer to these investigations as microprocess research. A major goal of this workshop is to suggest ways in which to integrate these two complementary lines of research to create a rigorous, orderly discipline of software process engineering. This integration could suggest, for example, how high level process behaviors might be predicted, and modified, through lower level analyses and optimizations. It could also explore how best to integrate objective microprocesses based on explicit knowledge with more subjective collaboration processes based on tacit knowledge.

The workshop succeeded in reaching its goal, namely to provide a forum for assessing current and emerging software process capabilities with respect to the challenges, and for obtaining insights into the software process research directions needed to address the challenges and make progress toward overriding goals. It included initial presentations by leading international software process researchers and users, presentations of contributed papers on process challenge areas and solution approaches, tool demonstrations, and a closing panel on software process research directions.

In response to the call for papers, 111 submissions were received from 10 different countries and regions: Australia, Canada, China, France, Germany, Hong Kong, Japan, New Zealand, UK and USA. Every paper was rigorously reviewed and held to very high quality standards, and finally 30 papers were accepted as regular papers for presentation at the workshop, representing a 27% acceptance rate for regular papers. In addition, 18 were selected as poster papers.

2 Keynote Addresses

The SPW2005 program was highlighted by 11 keynote speeches as listed in Table 1 (in alphabetic order by speakers' surname).

Table 1 Keynote speeches in SPW2005

Speaker	Topic	Affiliation
Victor R. Basili	Evolving Defect 'Folklore': A Cross-Study Analysis of Software Defect Behavior	University of Maryland, USA
Barry W. Boehm	The Future of Software Processes	University of Southern California, USA
Jacky Estublier	Software are Processes Too	French National Research Center in Grenoble, France
Watts S. Humphrey	Software: A Paradigm for the Future	Software Engineering Institute, Carnegie Mellon University, USA
Ross Jeffery	Achieving Software Development Performance Improvement Through Process Change	University of New South Wales, Australia
Mingshu Li	Expanding the Horizons of Software Development Processes: A 3-DIntegrated Methodology	Institute of Software at the Chinese Academy of Sciences, P.R.China
Leon J. Osterweil	Integrating Microprocess and Macroprocess Software Research	University of Massachusetts Amherst, USA
Arthur Pyster	What Beyond CMMI is Needed to Help Assure Program and Project Success?	Science Applications International Corporation, USA
H. Dieter Rombach	Integrated Software Process & Product Lines	Fraunhofer IESE & University of Kaiserslautern, Germany
Wilhelm Schäfer	A Rigorous Software Process for the Development of Embedded Systems	University of Paderborn, Germany
Brian Warboys	Active Models: A possible approach to the integration of objective and subjective process models	University of Manchester, UK

Victor R. Basili makes a cross-study analysis of software defect behavior. Answering "macro-process" research issues—which require understanding how development processes fit or do not fit in different organizational systems and environments—requires families of related studies. While there are many sources of variation between development contexts, it is not clear a priori what specific variables influence the effectiveness of a process in a given context. These variables can only be discovered opportunistically, by comparing process effects from different environments and analyzing points of difference. This paper illustrates the approach and the conclusions that can be drawn by presenting a family of studies on the subject of software defects and their behaviors—a key phenomenon for understanding macro-process issues. During his presentation, Basili also summaries some topics of "Empirically

Evolving Software Engineering Techniques”. Barry W. Boehm highlights the future of software processes. In response to increasing demands being put onto software-intensive systems, software processes will evolve significantly over the next two decades. His paper identifies seven relatively surprise-free trends—increased emphasis on users and end value; increasing software criticality and need for dependability; increasingly rapid change; increasingly complex systems of systems; increasing needs for COTS, reuse, and legacy software integration; and computational plenty—and two “wild card” trends: increasing software autonomy and combinations of biology and computing; and discusses their likely influences on software processes between now and 2025. It also discusses limitations to software process improvement, and areas of significant software process research and education needs.

Jacky Estublier discusses new dimensions of processes, at the light of the MDA framework, and shows what the differences and synergies are. A process defines the way activities are organized, managed, measured, supported and improved to reach a goal. It has been shown that “software processes are software too”; more precisely that their description can also be software. His paper hypothesizes that a system can be characterized by its goal and by answering the questions: why, what and how. It shows that software process work investigated only a tiny subset of processes, where only the how have been addressed. “Meta-process” research tried to address the why to change a process model, but was largely unfruitful. This paper first relates processes, software production and humans in the framework of the meta pyramid proposed by the OMG MDA. It shows that programs and process models are fully similar, but not at the same level in the meta pyramids. Therefore the claim: software are processes too. The meta pyramid framework is used to show and contrast new and original potential uses of process technology. Particularly strategic software management requires a kind of process support where the what is not humans, but the software itself. Finally autonomic computing will soon require process support where the why, the what and the how will have to be fully formalized and the process models automatically executed.

Watts S. Humphrey presents a future software paradigm. The software industry appears to have reached a plateau. While improvements have been made in the last 20 years, progress has been limited in scope and degree. Researchers, tool and method developers, and process specialists are all doing creative and promising work. However, as we continue making impressive technical and process advances, and even though occasional projects produce extraordinary results, broad and effective use of even generally-available best processes and methods has been slow and limited. Where new processes and methods have been properly introduced, the results have generally been positive. This would be considered success by many definitions but most software work continues to be done with ill-defined processes, poor tools and methods, and ineffective management systems. Modern software work involves many topics and there are many different specialties. Because our field is now so broad and so many different topics are now important, we have developed a wide variety of disciplines which each has its own experts. Unfortunately, these experts all have different views and, because we don’t agree among ourselves on what is important, our story is incoherent. Watts emphasized that to improve software industry so that it can meet the growing demands of society, the software process research community, must develop a coherent, consistent, and forceful position.

Ross Jeffery explores the context in which software process improvement occurs and the topic of process innovation. His paper summarizes the results of process improvement activities in two small software organizations. One of these made use of macro process modeling. These results, along with the reported results of CMMi adoption, are interpreted in the light of organizational theory, a process improvement research framework, and process innovation theory. It is concluded that the evidence supports process innovation or variations on innovation as a means of achieving large-scale improvements in productivity or quality. It also argues: (1) for the use of the process research framework to identify research limitations; (2) that consideration of process alone is unlikely to provide sufficient evidence for generalization.

Mingshu Li investigates how to define and improve software development processes. Based on examining the software development during last two decades, his paper proposes a breakthrough point of an updated view of requirements, called Great Requirements, and presents a 3-D Integrated Software Engineering methodology for improving software development activities. It expands the horizons of possible future software development processes.

Leon J. Osterweil proposes the unification of two complementary approaches to software process research. The two approaches can be characterized as macroprocess research, focused on phenomenological observations of external behaviors of processes, and microprocess research, focused on the study of the internal details and workings of processes. His paper suggests that it is time to bring these approaches together with the goal of using microprocess methods to provide definitive explanations of observed macroprocess behaviors. The paper suggests that this unification could lead to improved understandings leading to improvements in software development practice. The paper observes that such positive outcomes have resulted when the macro—and micro—approaches have been synthesized in domains such as Economics, Physics, and the Life Sciences.

Arthur Pyster examines aspects beyond CMMI that are needed to help assure program and project success. The U.S. Department of Defense and other parts of the U.S. government use the Capability Maturity Model Integration (CMMI) for process improvement to reduce the risk of poor performance by its major contractors. Acquisition officials have reported that many of its major programs suffer from cost, schedule, and technical performance problems even though those programs are being implemented by companies which rate high with respect to the CMMI. His paper explores possible reasons why companies with high CMMI ratings can still have significant performance problems and suggests possible remedies.

H. Dieter Rombach suggests an expansion of the well-defined and proven principles of software product line engineering to processes, and an integration of both based on the ideas of the "Experience Factory". Increasing demands imposed on software-intensive systems will require more rigorous engineering and management of software artifacts and processes. Software product line engineering allows for the effective reuse of software artifacts based on the pro-active organization of similar artifacts according to similarities and variances. Software processes—although also variable across projects—are still not managed in a similar systematic way. This paper motivates the need for Software Process Lines similar to Product Lines. As a result of such organization, processes within an organization could be organized according to similarities and differences, allowing for better tailoring to specific project needs (corresponds to application engineering in product lines). The vision of SPPL (integrated software process & product lines) engineering is presented, where suitable artifacts and processes can be chosen based on a set of product & process requirements and project constraints. The paper concludes with some resulting challenges for research, practice, and teaching.

Wilhelm Schäfer reports about a rigorous software process for embedded system development. The process is based on specifying the software by a well-chosen and well-defined subset of UML diagrams. The process together with a formally defined semantic of the various diagram types supports requirements tracking, consistency checking and formal verification across the various parts of an UML-Based specification. An existing software development environment illustrates the concepts of the paper.

Brian Warboys points out that we need to not only address the contexts for so called macroprocess and microprocess definitions in order to integrate them, but also address the underlying software engineering paradigm that currently constrains system designers to seek to suppress emergent behaviour. His paper suggests that the workshop problem of managing the integration of processes based on both explicit and tacit knowledge needs to be addressed by questioning the classical software engineering paradigm. It illustrates a possible approach through a short description of the recently prototyped ArchWare system.

3 Regular Papers

The SPW2005 consisted of five regular sessions of "Process Content" (8 papers), "Process Tools and Metrics" (4 papers), "Process Management" (4 papers), "Process Representation and Analysis" (7 papers), "Experience Reports" (7 papers), and a poster session (18 papers).

3.1 Process content

Sutton Jr. introduces Aspect-Oriented Software Development (AOSD) and presents the AOSD implications for software products, processes and process languages. He suggests a middle layer, mesoprocess, to unify the macroprocess and microprocess layers.

Yang provides three levels of process patterns, lifecycle patterns, activity patterns, and workflow patterns, to a general COTS-Based applications (CBA) process framework.

Huang proposes a Value-Based Software Dependability Achievement (VBSDA) process generated from the WinWin Spiral Model's risk-driven approach.

Song, Rudorfer, Hwong, Matos, and Nelson define a highly concurrent, software rapid prototyping process (S-RaP) in Siemens that supports a sizable development team to develop a high-quality, evolutionary software prototype.

Huang, L.Yang, and Y.Yang establish a set of measurable assessment criteria in assessing open source software candidates and making optimal decisions in the development process.

Mao, Lu, and Wang propose an Artificial Neural Network (ANN) method to assist the requirements changes' distribution and cost prediction.

Tian, Zhang, Zhou, and Qian propose a gradually proceeded software architecture design process (GADesign).

Ning, Hou, Hua, Yu, and Hao suggest a systematic view for requirement process improvement.

3.2 Process tools and metrics

Harada, Awane, Inoya, Ohno, Matsushita, Kusumoto, and Inoue develop a project management system, PRO-NAVI, based on their Work-Breakdown-Structure Process Model.

Ding, Yang, Sun, Tong, and Wang propose an evaluation framework for the capability of PSP based on Data Envelopment Analysis (DEA).

Wang and Meyers propose a framework named Spiral Pro that integrates Spiral Model, MBASE and COCOMO II to help project managers do their project planning in a systematic way.

Chen, Ying, Xue, and Zhao define an exact mapping from UTP (UML Testing Profile) to TTCN-3 and present a two-step transformation method leading to an automation or semi-automation of software testing process.

3.3 Process management

Wang and Li examine the current status of software process management in China and present a promising solution that transfers the software process improvement problems into quality management, development technology and services support. It also develops a toolkit, called SoftPM and adopted widely in China, to help software organizations improving their software processes.

Nejmeh and Riddle present a process evolution dynamics framework based on an experience-based categorization of process evolution-related activities highlighting their collaborative maturation of a company's process knowledge base.

Gong, He, and Liu put forwards a process improvement framework oriented to Chinese small organizations.

Kindler, Rubin, and Schäfer present an idea that exploits the user interaction with a version management system for the incremental automatic derivation, refinement and analysis of process models. It also sketches the architecture of the solution and the algorithms for the main steps of incremental automatic derivation of process models.

3.4 Process representation and analysis

Osterweil, Wise, Clarke, Ellison, Hadley, Boose, and Foster introduce the concept of an analytic web, a synthesis of three complementary views of a scientific process that is intended to facilitate the conduct of science. It also describes experiences with a tool, SciWalker, designed to evaluate the efficacy of this approach. SciWalke provides 3 notations, dataflow graph, data derivation graph and Little-JIL process definition and incorporates the first two currently. Its application already has led to new scientific insights and interesting new results.

Klappholz and Port present an upward-tailorable process wrapper framework for identifying and avoiding model clashes, called M(in)BASE, a minimal version of MBASE.

Raunak and Osterweil describe how the features of the Little-JIL process definition language helped in the rapid generation of simulations that shed important new light on the effectiveness of various collusion strategies in influencing the outcomes of various auction approaches.

Madachy uses modeling and simulation to assess process tradeoffs for business case analysis and shows how software business decision-making can improve with value information gained from simulation models that integrate business and technical perspectives.

Bhuta, Boehm, and Meyers discuss the duality between product and process reuse. It proposes the development of process elements, “process counterparts to software components”, which can be built with reusable strategies and then be integrated with other process elements to develop software plans.

Ge, Hu, Lu, Hu, and Lü use nets within nets to model cross-organizational software processes based on mobile agent systems. It presents translation rules from nets within nets to flat nets, which preserve the soundness property.

Clarke, Y.Chen, Avrunin, B.Chen, Cobleigh, Frederick, Henneman, and Osterweil make a successful case study on the Medical Safety Project at the University of Massachusetts. It uses the process programming language Little-JIL to specify a real-world, non-trivial in patient blood transfusion process and verifies the process to satisfy some important safety properties.

3.5 Experience reports

Jensen and Scacchi share their experiences in discovering, modeling and reenacting processes associated with large, global open source software development (OSSD) projects like NetBeans.org.

Kuhrmann, Niebuhr, and Rausch present some experiences in application of the V-Modell XT for the German Department of the Interior pilot project “Development of the WiBe software”.

Chen, Port, Chen, and Boehm apply the Experience Factory approach to collect and evolve a large amount of software process experience into an experience base (eBASE) through many real-client project software engineering practices at the USC Center for Software Engineering since 1996. The eBASE has been leveraged successfully for empirically based software process research and realized tangible benefits in automating, organizational learning, and strategic advantages for software engineering research.

Sugahara, Ogasawara, Aoyama, and Higashi report the status of Japanese SPI activities through their experiences in JASPIC (Japan SPI Consortium).

Podorozhny, Perry, and Osterweil describe their experiences in a logistics software process from the telecommunication domain using the Little-JIL based Artifact-based Analysis (ABA) automated assistance.

Lui and Chan propose an implementation roadmap that shows how inexperienced software teams in industrial developing areas in China can adopt eXtreme Programming (XP) to produce software applications.

Wu, Christensen, Li, and Wang make a survey of CMM/CMMI implementation in China. It identifies the reasons, success factors and benefits, and analyzes the problems in CMM/CMMI usage, by investigating most of the organizations which have been appraised. It also gives some recommendations to the Chinese software industry and government to solve those problems, which could help those who want to or are using CMM/CMMI.

4 Software Development Support Tools Demonstration

8 software development support tools were demonstrated in the workshop. The tools and vendors are listed in Table 2.

Table 2 Tools demonstrated in SPW2005

Tool	Vendor
Cost Xpert Project Estimation Tool	Cost Xpert Group, USA
Spiral Pro	Software Process Group, Inc., USA
Mobile Tools for Requirements Discovery	Johannes Kepler University Linz, Austria
Risk Assessment and Tracking System	RATS Software Research Associates Inc., Japan
Concern-Based Business Process Modeling	IBM China Research Laboratory, China
Performance Testing Tool for Wireless Applications	Hong Kong Polytechnic University
SoftPM (Integrated Software Process Services Management System)	Institute of Software, Chinese Academy of Sciences, China
UDCORE(User-driven Domain-specific COmponent-Based Requirements Elicitation tool)	Institute of Software, Chinese Academy of Sciences, China

5 The Panel “Where Are We Now? Where Should We Go Next?”

Chaired by Prof. Leon J. Osterweil, SPW2005 ended with a closing panel on the discussion of the future directions for software process research: “Where Are We Now? Where Should We Go Next?” The panelists included: Prof. Barry Boehm, Prof. Mingshu Li, Prof. Ross Jeffery, and Prof. Wilhelm Schäfer, representing SPW 2005 participants from America, Asia, Australia and Europe respectively.

There were 235 registered participants, 50 were from America, Europe, Australia, and other Asian countries outside China. The others were from different Chinese cities, such as Beijing, Shanghai, Nanjing, Xi’an, Wuhan, Hangzhou, Changsha, Chengdu, Changchun, Guangzhou, Shenyang, and they covered most of the best universities and research organizations in China.

For any more information, please visit the SPW 2005 website at http://www.cnsqa.com/~spw2005.

Acknowledgement A conference like this can only succeed as a team effort. All of this work would not have been possible without the dedication and professional work of many colleagues. We wish to express our gratitude to all contributors for submitting papers. Their work forms the basis for the success of the workshop. We also would like to thank the program committee members and reviewers because their work is the guarantee for the high quality of the workshop. Particularly thanks also go to the keynote speakers for giving their excellent presentations in the workshop.

References:

[1] Fuggetta A. Software process: A roadmap. In: Fuggetta A, ed. The Future of Software Engineering 2000: 22nd Intl. Conf. on Software Engineering. ACM Press, 2000. 25–34.

[2] Pfleeger SL. Software Engineering: Theory and Practice. 2nd ed, Prentice-Hall, 2001.

[3] Boehm BW. A spiral model for software development and enhancement. IEEE Computer, 1998,21(5):61–72.

[4] Osterweil LJ. Software processes are software too. In: Proc. of the 9th Int'l Conf. on Software Engineering (ICSE 9). ACM Press, 1987. 2–13.

Mingshu Li, program cochair of SPW 2005, is the Research Professor in the Lab for Internet Software Technologies and the State Key Lab of Computer Science, Institute of Software at the Chinese Academy of Sciences. He is the Director of the Institute from 2002. He is the Member of the Information Technology Steering Committee for the National Hi-Tech Research and Development Program and the Standing Member of the Chinese Computer Federation. He received his Ph.D. degree from the Department of Computer Science, Harbin Institute of Technology in 1993; also a Master degree (secondary one) in Economics from Department of Social Sciences at the same university in 1995. His current research interests are in software engineering (particularly requirement engineering and software process), real-time systems and Internet/Web technologies. He has served as the Editor-in-Chief of the Journal of Software. Contact him at the Inst. of Software, Chinese Academy of Sciences, P.O.Box 8718, No.4 South Fourth Street, Zhong Guan Cun, Beijing, China 100080; mingshu@iscas.ac.cn.

Barry W. Boehm, program cochair of SPW 2005, is the TRW Professor of Software Engineering and director of the Center for Software Engineering at the University of Southern California. His research interests include software process modeling, software requirements engineering, software architectures, Software metrics and cost models, software engineering environments, and value-based software engineering. His contributions to the field include the Constructive Cost Model (COCOMO), the Spiral Model, and the Theory W (win-win) approach to software management and requirements determination. He is a fellow of the ACM, AIAA, IEEE, and INCOSE and is a member of the US National Academy of Engineering. He received his Ph.D. degree from UCLA in mathematics. Contact him at the Center for Software Eng., Univ. of Southern California, 941 W. 37th Place, SAL Room 337, Los Angeles, CA 90089-0781; boehm@cse.usc.edu.

Leon J. Osterweil, program cochair of SPW 2005, is a professor in the Department of Computer Science at the University of Massachusetts, Amherst. Previously, he was a professor in, and chair of, the Computer Science Departments at both the University of California, Irvine, and the University of Colorado, Boulder. He was the founding director of the Irvine Research Unit in Software (IRUS) and the Southern California SPIN. He has been program committee chair of ICSE 16, TAV 2, ISPW4, and SDE2; and general chair of FSE 6. He has also presented keynote talks at such meetings as CASE '92 in Montreal, Quality Week '96 in San Francisco, the Inaugural Symposium of JAIST (the Japan Advanced Institute for Software Technology) in Kanazawa, Japan, and ICSE 9 (the Ninth International Conference on Software Engineering (ICSE 9) where he introduced the concept of process programming. His ICSE 9 paper was awarded a prize as the most influential paper of ICSE 9, honored as a 10-year retrospective. He has consulted for such organizations as IBM, Bell Labs, SAIC, MCC, and TRW, and is a member of SEI process program advisory board. Professor Osterweil is a fellow of the ACM. Contact him at the Department of Computer Science, University of Massachusetts, Amherst MA 01003; ljo@cs.umass.edu.