

**PROCEEDINGS OF THE INTERNATIONAL
CONFERENCE ON SOFTWARE ENGINEERING
RESEARCH AND PRACTICE**

SERP'03

Volume I

Editors:

Ban Al-Ani

H. R. Arabnia

Youngsong Mun

Associate Editors:

**J. L. R. Becerra, L. Chung, H. A. X. Costa,
S. Dascalu, E. Grant, C. Juiz, A. Korthaus,
H. Reza, M. Schader, N. Subramanian**

Las Vegas, Nevada, USA

June 23-26, 2003

©CSREA Press

This set of volumes contains papers presented at the International Conference on Software Engineering Research and Practice (SERP'03). Their inclusion in this publication does not necessarily constitute endorsements by the editors or by the publisher.

Copyright © 2003 by CSREA Press

Copyright and reprint permission:

Copying without a fee is permitted provided that the copies are not made or distributed for direct commercial advantage, and credit to the source is given. Abstracting is permitted with credit to the source. Contact the editors or the publisher, for other copying, reprint, or republication permission.

Volume I	ISBN: 1-932415-19-X
Volume II	ISBN: 1-932415-20-3
Set	ISBN: 1-932415-21-1

Printed in the U.S.A.

Foreword

It gives us great pleasure to introduce this collection of papers to be presented at the 2003 International Conference on Software Engineering Research and Practice (SERP'03), June 23 through 26, 2003, at Monte Carlo Resort, Las Vegas, Nevada, USA. The SERP'03 conference is co-sponsored and organized by Computer Science Research, Education, and Applications Press (CSREA); International Technology Institute (ITI); Korean Society for Internet Information (KSII); The World Academy of Sciences for Information Technology (WAS); and a number of computer science book publishers, usersgroups and newsgroups.

The program committee would like to thank all those who submitted papers for consideration. About 40% of the submissions were from outside the United States. Each submission was evaluated by two referees (except for papers that were directly submitted to chairs of sessions who were responsible for the evaluation of these papers.) The overall paper acceptance rate was about 39%.

We are very grateful to the many colleagues who helped in organizing the conference. In particular, we would like to thank the members of the SERP'03 Program Committee who we hope will offer their help again in organizing the next year's conference (SERP'04). The SERP'03 Program Committee members are:

Dr. Hamid Abachi, Monash University, Victoria, Australia
Dr. Ban Al-Ani (Vice Chair), University of Technology, Sydney, Australia
Dr. Shereef Abu Al-Maati, State University of West Georgia, USA
Prof. Yamine Ait Ameer, University of Poitiers, France
Dr. H. R. Arabnia (Co-Chair), University of Georgia, GA, USA
Dr. Jorge L. R. Becerra, USP, Brazil
Dr. Luiz Ricardo Begosso, FEMA Assis, Brazil
Dr. Asoke K. Bhattacharyya, Saint Xavier University, Chicago, IL, USA
Dr. Chia-Chu Chiang, University of Arkansas at Little Rock, AR, USA
Dr. Li Kuan Ching, Providence University, Taiwan
Prof. William Chu, TungHai University, Taiwan
Prof. Lawrence Chung, University of Texas at Dallas, Richardson, USA
Prof. Heitor A. X. Costa, Federal University of Lavras, Brazil
Dr. Sergiu Dascalu, University of Nevada, Reno, USA
Dr. Mourad Debbabi, Panasonic Inf. & Net. Tech., Princeton, NJ, USA
Prof. Filomena Ferrucci, Universite di Salerno, Italy
Prof. Jinan A. W. Fiaidhi, Lakehead University, Canada
Dr. E. Grant, University of North Dakota, Grand Forks, ND, USA
Prof. John Grundy, University of Auckland, New Zealand
Prof. Karim El Guemhioui, Universite du Quebec en Outaouais, Canada
Dr. Jiang Guo, California State University Los Angeles, CA, USA
Dr. Carlos Juiz, Universitat de les Illes Balears, Spain
Jan Jurjens, Munich University of Technology, Germany
Dr. Marcel Karam, American University of Beirut, Lebanon
Tahar Khammaci, IRIN Universite de Nantes 2, France
Dr. Anil Khatri, United States Patent and Trademark Office, USA
Dr. Paddy Krishnan, Bond University, Australia
Dr. Axel Korthaus, University of Mannheim, Mannheim, Germany
Prof. Youngsong Mun (Co-Chair), SSU, Korea
Thomas Panas, Vaxjo University, Vaxjo, Sweden
Prof. Habil Ilka Philippow, Technical University of Ilmenau, Germany
Prof. Bhanu Prasad, Georgia State University, Georgia, USA
Prof. Corrado Priami, University of Trento, Italy
Dr. Muthu Ramachandran, Leeds Metropolitan University, Leeds, UK
Dr. Samira Sadaoui, University of Regina, Canada

Prof. Martin Schader, University of Mannheim, Mannheim, Germany
Dr. Trevor Smedley, Dalhousie University, Halifax, Canada
Dr. Nary Subramanian, Anritsu Company, Richardson, Texas, USA
Professor K.-Y. Sung, Handong University, Korea
Ladan Tahvildari (Student Member), University of Waterloo, Canada
Dr. Gregory Vert, University of Nevada, Reno, USA
Prof. Houman Younessi, Rensselaer Polytechnic Institute, USA
Dr. Andrea Zisman, City University, London, UK
Dr. Raed Abu Zitar, Philadelphia University, Amman, Jordan
Members of MultiConference Task Force for Software Engineering
Members of World Academy of Sciences Task Force for Information Technology

We would also like to thank the following: Prof. H. J. Siegel for his continued support for the MultiConference, presenting a tutorial and the MultiConference Keynote; Dr. Tim Field and Mr. Chan Field of APC for managing the printing/publication of the proceedings; April Turner, Brooke Gray and Janice Whitehead-Rhodes of APC for meeting publication and printing deadlines; Prof. Iyad Ajwa for maintaining the conference web sites; The staff of Monte Carlo Resort in Las Vegas (conference division); The staff of Universal Conference Management Systems and Support (UCMSS, San Diego, California) for the professional service they offered us, in particular, thanks to Tom Scrivner and Bobbie Carter. Last but not least, we would like to thank the SERP's Associate Editors: Jorge L. R. Becerra, Lawrence Chung, Heitor A. X. Costa, Sergiu Dascalu, E. Grant, Carlos Juiz, Axel Korthaus, Hassan Reza, Martin Schader and Nary Subramanian.

We present the proceedings of SERP'03.

H. R. Arabnia, Ban Al-Ani and Youngsong Mun
Co-Chairs, SERP'03 Program Committee

Contents

Volume I

Session: Using the J2EE Platform for Efficient Component-Oriented Development of Enterprise Systems (JEFF-CODES)

Integrating J2EE Services into a Heterogeneous Service Environment by Means of Model Driven Engineering	3
Tom Ritter, Klaus-Peter Eckert and Andre Szdzuy	
Efficient Component-Based Integration of a Mobile J2EE Application with a Java-Based "Legacy" System – A Best-Practice Example	10
O. Höß, A. Weisbecker, U. Rosenthal and M. Veit	
Transitioning from an "EAI Architecture" to a "Decoupling Architecture": The J2EE Solution	16
Denis Dorssers, Stefano Martinotti and Andrew Tay	
User Interface Code Generation for EJB-Based Data Models Using Intermediate Form Representations	23
Branko Milosavljević, Milan Vidaković, Srdjan Komazec and Gordana Milosavljević	
Handling Huge Data Sets in J2EE/EJB2.1 with a Page-by-Page Iterator Pattern Variant for CMP	28
Ralf Gitzel, Axel Korthaus and Nima Mazloumi	
A Critical Analysis of JDO in the Context of J2EE	34
Axel Korthaus and Matthias Merz	

Session: Software Performance Engineering

Exploring Roles for the UML Diagrams in Software Performance Engineering	43
José Merseguer and Javier Campos	
Extending UML Deployment Diagrams form a Performance Engineering Perspective	48
Juan-Luis Anciano and R. Puigjaner	
Performance Assessment of Software Models in a Configurable Environment Simulator	55
Michael Barth	
A Software Performance Evaluation Approach Using Stochastic Process Algebra Tools	62
Abdelmalek Benzekri and Osman Salem	
Developing a Formal Design Analysis Framework	68
Kendra Cooper, Lirong Dai, Yi Deng and Jing Dong	
MASCOTime: Building a Simulator with Java for Annotated MASCOT	74
Pere P. Sancho and Carlos Juiz	

A Performance Assessment Model for Distributed Software Systems	81
Debra L. Smarkusky	
Integrated Tools for Performance-Oriented Distributed Software Development ...	88
N. Mazzocca, E. Mancini, M. Rak and U. Villano	
Performance of Service-Discovery Architectures in Response to Node Failures	95
C. Dabrowski, K. Mills and A. Rukhin	

Workshop: Adaptable Systems/Software Architectures

Generative and Incremental Approach to Scripting Support Implementation	105
Vespe Savikko	
An Architecture for the Development, Evolution and Adaptation of Hypermedia Systems	112
Nuria Medina-Medina, Fernando Molina-Ortiz, M ^a José Rodríguez-Fórtiz and Lina García-Cabrera	
A Runtime Transformation Method for Distributed Systems	119
Masaki Murakami	
Measures for Mobile Users	126
Alberto Sillitti, Andrea Janes, Giancarlo Succi and Tullio Vernazza	
Relating Evolving Business Rules to Software Design	129
Wan M. N. Wan Kadir and Pericles Loucopoulos	
Assessing Systems Adaptability to a Product Family	135
Mika Korhonen and Tommi Mikkonen	
A Software System Evolutionary and Adaptive Framework. Application to Agent Based Systems	142
Patricia Paderewski-Rodríguez, Juan Jesús Torres-Carbonell, M ^a José Rodríguez-Fortiz, Nuria Medina-Medina and Fernando Molina-Ortiz	
Semi-Automatic Generation of Adaptable Architectures	149
Nary Subramanian and Lawrence Chung	
Architecting Adaptable Software Using COTS: An NFR Approach	155
Lawrence Chung, Kendra Cooper and Anna Yi	

Session: Software Engineering: Concepts, Issues and Applications

Software Development: Planning x Agility	165
Enrico de Sousa Visconti and Eidson Spina	
An Automatic Approach to Transform CafeOBJ Specifications to Java Template Code	171
C. Dounsa-ard and T. Suwannasart	
The ODP-UP Method to Define Open and Distributed Architectures	177
Marcelo Galasini and Jorge Luis Risco Becerra	
Integrating Aspects with Model Driven Software Development	186
Vinay Kulkarni and Sreedhar Reddy	
Java Code Generator Using UML and OCL	193
Djalma D. Silva and Edson S. Gomi	

A Portable and Collaborative Distributed Programming Environment	198
Chang-Hyun Jo and Allen J. Arnold	
A Process for BDI Agent-Based Software Construction	204
Chang-Hyun Jo and Jeffery M. Einhorn	
An ODP Vision of QoS Management: An Application in the SLM Architecture	210
Cristina Mori Miyata and Jorge Luis Risco Becerra	
Current Issues in Software Process Automation	217
Jiang Guo, Behzad Parviz and Raj Pamula	
The Development Programmatics of Large Scientific Codes	224
Jack K. Horner	
A New Multi-Tasking Concept Supported by SMV	228
Duc-Duy Vo and Claude Petitpierre	
Recommendations on Transition from CMM to CMMI	234
Mansour Zand and Raghunath Shapkota	
Design and Implementation of an Inter-Process Communication Model for an Embedded Distributed Processing Network	239
K. W. Choy, A. A. Hopgood, L. Nolle and B. C. O'Neill	
Process Improvement in Large-Scale Industrial Environments Based on SCM ...	246
Tilman Seifert, Markus Pizka and Karlheinz Raith	
Supporting Communication Practices: How the Choice of Methodologies Can Affect Requirements Elicitation	250
Jane Coughlan and Robert D. Macredie	

Session: Modeling Notations, Tools and Domains + Log Files ...

Reliability Modeling Using UML	259
Chokchai Leangsuksun, Hertong Song and Lixin Shen	
A Comparison of Software Quality Modeling Techniques	263
Justin M. Beaver and Guy A. Schiavone	
Extending Software Models Based on a Metamodel for Predicting Software Performance During Unified Software Development Process	267
Zhongfu Xu and Axel Lehmann	
An UML Case Tool with Compound Document Support	274
Taegyun Kim and Jang-Wu Jo	
Taking Advantage of the Symbiotic Relationship Between Tools and Processes to Support Executable Processes to Support Executable Process Models	279
Ralf Buschermöhle and Wilhelm Hasselbring	
A Bestiary of Log Files: Log File Format in Distributed Systems	286
H. Keith Edwards and Michael A. Bauer	
A Methodology for Assuring Quality Logging Practices in Electronic Commerce Systems	293
Michael A. Bauer and Don Bourne	
A Practical Approach to the Evaluation and Selection of CASE Tools	300
Hassan Pournaghshband, Mandana Sadigh and Shahriar Movafaghi	

Formal Collaborative Production Environment	305
Jia Zhang, Carl K. Chang, Francis K. H. Quek and Zhiguo Gong	
Business Model Driven Approach for Web Application Development	312
Yongsun Cho and Kiwon Chong	
A Design Method of Communication Protocols Using SDL Patterns	319
Ki-Sook Chung, Byung-Sun Lee and YoungJoon Byun	

Session: Software Architectures

A Framework for Specifying Software Architecture Based on Multi-Formalisms	325
Hassan Reza and Emanuel Grant	
Applying the IEEE 1471-2000 Recommended Practice to a Software Integration Project	332
Rikard Land	
A New Approach to Evolve Dependability and Performance for Software Architecture	339
Samir Benarif, Amar Ramdane-Cherif and Nicole Levy	
Toward a Generation of Code Multi-Target for the VBOOM Method	345
Boubker Sbihi, Kriouile Abdelaziz, Ettalbi Ahmed and Bernard Coulette	
Memory Access Characteristics of Network Infrastructure Applications	351
Abdul Waheed	
Computer Supported Cooperative Work Oriented Architecture	358
Jia Zhahg, Carl K. Chang, Kai H. Chang and Zhiguo Gong	
Measuring the Coherence of Software Product Line Architectures	364
Vojislav B. Mišić	

Session: Components Based Software Engineering

A Practical Component Testing Technique Based on Component Contract	373
YoungTaek Jin and SunMyung Hwang	
Component Adaptation Through Adaptation Components	379
Jeong Ah Kim	
A Framework for Domain-Specific Reusable Component	384
Hisham Haddad and Walter Fortner	
Component-Based Software Engineering: Issues and Concerns	391
Hisham M. Haddad and Erol Biberoglu	
3CoFramework: A Component-Based Framework for Distributed Applications	398
Shifeng Zhang and Steve Goddard	

Sessions: Requirements Engineering

Extending Acquisition of High Quality Customer Level Requirements	407
Rattikorn Hewett, John Leuchner, Ken Ford and Dan Cooke	

Usability and Adaptability of Evaluation Framework for Requirements Engineering Tools	414
Raimundas Matulevičius	
A Software Engineering Process to Specify and Verify E-Commerce Systems	419
M. Song, A. Pereira, F. Lima, G. Gorgulho, S. Campos and W. Meira	
Sowing the Seeds of Requirements Engineering in Post-Graduate Students	426
Ban Al-Ani	
Critical Success Factors for the Improvement of Requirements Engineering Process	433
Mahmood Niazi and Sudha Shastry	
Requirements Engineering for Application Development in Volatile Project Environments Using Continuous Quality Function Deployment (CQFD)	440
Georg Herzwurm, Sixten Schockert, Ulrike Dowie and Michael Breidung	
Tracing Software Requirements Artifacts	448
Andrea Zisman, George Spanoudakis, Elena Pérez-Miñana and Paul Krause	
An Approach for the Synthesis of State Transition Graphs from Use Cases	456
Stéphane S. Somé	

Contents

Volume II

Session: Software Specifications: Techniques, Tools and Case Studies

Directions of Using UML for Software Specification: An Overview	465
Manish Nilawar and Kedar Deshpande	
A Case Study in Object Oriented Modeling, Architecting, and Designing an Enterprise Monitoring Application	470
Sushil J. Louis, John McDonnell, Doan Hohmeyer, Lisa Heinselman and Andrew Walker	
Specification of the Verity Learning Companion and Self-Assessment Tool	476
Sergiu Dascalu, Daniela Saru, Ryan Simpson, Justin Bradley, Eva Sarwar and Joohoon Oh	
Software Requirements Specification of a University Class Scheduler	490
Deanna M. Needell, Jeff A. Stuart, Tamara C. Thiel, Sergiu Dascalu and Frederick C. Harris, Jr.	
Component Framework for Visual Programming Language Design	497
Simon Gauvin and Trevor Smedley	
An Abstract Graph Model to Collect Interprocedural Dataflows in Visual Dataflow Programs	503
Marcel R. Karam and Trevor Smedley	

Software Specification of A Mining Truck Simulator and Trainer	509
Frederick C. Harris Jr., Yan W. Ha, Dianne M. Yumul, Joshua S. Estes and Christopher E. Miles	
Specification of an Online Advisement System	516
Christian Rayburn, James Hays, Bryan Phillips and Frederick C. Harris Jr.	
Modeling and Formal Verification of IMPP	522
Sohel Khan and Abdul Waheed	
FAR: An Editing Tool for Standard Information Generation	529
Tao Zhang, H. Conrad Cunningham and Jian Li	
A Software Library for SyncML Server Applications	533
JiYeon Lee and Hoon Choi	
Web Task Timing for Web Project Cost Estimation	538
Kerri Korschgen and Hossein Tahani	

Session: Object Oriented Technology + OCL

Consistent Object-Oriented Modeling of System Dynamics with State-Based Collaboration Diagrams	547
Cornelia Heinisch and Joachim Goll	
Practical Use of Encapsulation in Object-Oriented Programming	554
Mats Skoglund	
Automating Object-Oriented Software Refactoring	561
Subash Shankar and Xiaowei Xu	
Mappings between Object-Oriented Technology and Architecture-Based Models	568
Peter Tabeling and Bernhard Gröne	
Polymorphism in Object-Oriented Contract Verification	575
I. Nunes	
An Application of Object Oriented Paradigm (OOP) to Combination Logic Design	582
S. Chaiworawitgul, P. Pitsatorn and B. Sowanwanichakul	
DPET – A Simple C++ Design Pattern Extraction Tool	588
Samuel A. Ajila and Peng Xie	
Expressing Property Specification Patterns with OCL	595
Stephan Flake and Wolfgang Mueller	

Session: Software Quality + Testing + Correctness

Analysis of Relationship Among ISO/IEC 15504, CMM, and CMMI	604
Sun-Myung Hwang	
Software Evaluation by User Satisfaction Analysis Based on Quality Characteristics for ISO/IEC 9126	610
Wonil Kwon, Hyo-Ri Jeon, Chang-shin Chung, Seokkyoo Shin and Insub Cho	
An Evaluation Model for Software Quality Improvement	615
Jae-kyu Cho and Sung Jong Lee	

Predicting Faulty Classes Using Design Metrics with Discriminant Analysis	621
Mathupayas Thongmak and Pornsiri Muenchaisri	
Adapting Function Point Analysis to Feature Models for Measuring Reusability	628
Wonseok Chae	
A Study on the Measurement for Embedded Software	635
Sang-Pok Ko, Kang-Tae Kim, Hyun-Dong Lee and Kyung-Whan Lee	
How to Verify Software Product Quality in User's View	638
Namhee Kim, Seokkyoo Shin and Insub Cho	
State of the Art End-Of-Time Tester	644
Mihyar Hesson	
A Maturity Model for the Implementation of Software Process Improvement	650
Mahmood Niazi and David Wilson	
Formal Specification Based Software Testing: An Automated Approach	656
Mandeep Singh Gill and Rajesh Kumar Bhatia	
Meta-Validation of UML Diagrams Using OCL Rules	660
Il-Kyu HA and Byung-Wook Kang	
Implementing ISO/IEC 12207 Standard Using Rational Unified Process	667
Sheila S. Reinehr, Ricardo Balduino, Cristina Â. JF. Machado and Marcelo S. Pessôa	
Formalization and Automated Testing for Cursive Fonts	681
Siamak Rezaei	
Test Plan Design for Software Configuration Testing.....	686
Baowen Xu, Changhai Nie, Liang Shi, William C. Chu, Hongji Yang and Huowang Chen	
ISO/IEC 15504 Adaptation for Software Process Assessment in SMEs	693
Antonia Mas Picahaco and Esperança Amengual Alcover	

Session: Implementation and Users Issues + Software Maintenance

Automatic Benchmarking and Optimization of Codes: An Experience with Numerical Kernels	701
José R. Herrero and Juan J. Navarro	
Putting Interdisciplinary Software Engineering into Practice: A Java-Based Embedded System Controller	707
D. Needham, M. Simpson and B. Whitten	
Multi User Monitoring and Real-Time Control with ST-RTL	714
R. Murillo Garcia, D. K. Harrison and B. G. Stewart	
InterCase: An Environment for Prototyping User Interfaces	721
Cristina Paludo Santos and Denílson Rodrigues da Silva	
Generation of User Interface Prototype for the Support of Usage-Centered Design	726
Jeong-Ok Kim, Cheol-Jung Yoo, Yong-Sung Kim and Ok-Bae Chang	
Formal Validation of HCI User Tasks	732
Yamine Ait Ameer, Mickael Baron and Patrick Girard	

Experiences Developing an E-Whiteboard-Based Circuit Designer	739
Ray Liu, Lisa Wong and John Grundy	
Investigating Software Design Measures as Indicators of Understandability	745
Subhas Chandra Misra	
Understanding the Impact of Change in COTS-Based Systems	752
John Hutchinson, Gerald Kotonya, Benoit Pete Sawyer	

Session: Software Engineering Methods and Environments

A Design Experiment for Software Engineering Curriculum	761
Xiaohong (Sophie) Wang and Willis S. Boyd	
Rigorous Software Engineering	766
Andrija Maricic	
Evolving Software Development Instruction to Support Agile Practices	773
Michael Wainer and Denny Hays	
Form-Based Object Analysis Process by Applying Reverse Engineering in Legacy Application Systems	780
Chang-Mog Lee, Cheol-Jung Yoo, Ok-Bae Chang and In-Su Kim	
A Method of Software Development Using BizWiz that Supports Automated Generation Systems	785
Hyoun-Goun Han, Myung-Jin Lee and Byung-Ug Kang	
Towards a Software Engineering Approach to Wireless Application Development	791
Qusay H. Mahmoud	
Creating an Environment for a High Performance Software Engineering Teams – A Real Life Perspective	798
Victor A. Clincy	
A Strategic Approach for Decision Making in Process Centered Software Engineering Environment	803
Inés Bayouth Saâdi, Yassine Jamoussi and Henda Ben Ghezala	
An Experience Report on Teaching the Personal Software Process	810
Xiaohong Yuan	
Open Giant Intelligent Information Systems and Its Multiagent-Oriented System Design	816
Longbing Cao, Chungsheng Li, Chengqi Zhang and Ruwei Dai	
Comparison of Industry-Sited Projects and University-Sited Projects for Final Year Students	823
E. Chang, C. Coleman, N. Jayaratna, W. Ye and M. Miller	

Session: Software Reuse + Reverse Engineering + Tools

Software Component Reuse Using Formal Methods & k-nn Technique	835
Simarjot Singh and Rajesh Kumar Bhatia	
Software Reuse: An Overview	840
Samira Sadaoui, Angela Mlynarski and Elspeth Nickle	

A Survey of Data Mining Technology Applied to Software Reuse	847
S. Tangsripairoj and M. H. Samadzadeh	
Towards the Unified Recovery Architecture for Reverse Engineering	854
Thomas Panas, Welf Löwe and Uwe Aßmann	
Building Software Via Shared Knowledge	861
José R. Herrero and Juan J. Navarro	

Session: Semantics

LEARN: An Alternative Formal Semantics Definition	871
George Karakitsos, Eleni Berki and Elli Georgiadou	
On Execution Semantics of UML Statechart Diagrams Using the π-Calculus	877
Vitus S. W. Lam and Julian Padget	
Formal Semantic Specification for a Set of UML Diagrams	883
Cui Zhang	
A Vision for Product Traceability Based on Semantics of Artifacts	890
Darijus Strašunskas	
While Loop Demonic Semantics Monotype/Residual Style	896
Fairouz Tchier	

Session: Late Papers and Post-Conference Papers

Configuration Management in Distributed Measurement Systems	905
P. Mariño, C. Sigüenza, F. Poza, F. Vázquez and F. Machado	
Induction of Survivability in Rational Unified Process (Rup[®])	915
Shahid Hussain Abbassi, Muhammed Saeed and Faheem Ahmed	
Strategies for Modeling Software Architectures in Virtual Reality Systems	919
Rafael Capilla and Margarita Martínez	
Object Oriented Requirements Engineering for Multidiscipline Systems	925
Thomas J. Wheeler	
Reverse Engineering Methodology to Recover the Design Artifacts: A Case Study	932
Nadim Asif	
Software Engineering Practice: A Case of Scale Reduction – Mini Software Factory of the Genesis Incubator of Pato Branco	939
Paulo Roberto Bueno, Marisângela Pacheco Brittes and Gilson Fonseca	
Automating Feature-Oriented Domain Analysis	944
Fei Cao, Barrett R. Bryant, Carol C. Burt, Zhisheng Huang, Rajeev R. Raje, Andrew M. Olson and Mikhail Auguston	
Expressing Real-Time Constraints in OCL with High-Level Temporal Logic Operators	950
José M. Garrido	
Internet System Design for the Disabled Net	956
Miss R. Jesmin and Kevin Charles Lano	

Managing Software Engineering Projects through Internet Technologies	963
Ed Rodgers	
Mapping Caché Artifacts to Design Metrics Primitives	966
Vinayak Tanksale, Dolores Zage and Christopher Steele	
Measurements Used in Software Quality Evaluation	971
Bernard Wong	
Scenario-Based System Generation for Process Control	978
W. T. Tsai, L. Yu, R. Paul, A. Saimi, W. Song and Z. Cao	
Scenario-Based Software Architecture Modeling	
Using Message Sequence Charts	985
Gerardo Padilla, Cuautémoc Lemus and Miguel A. Serrano	
Evaluating the Potential for Integrating the OPEN and Tropos Metamodels	992
B. Henderson-Sellers, P. Giorgini and P. Bresciani	
An Agent-Based Collaborative Architecture for	
Knowledge-Driven Process Management	996
Aizhong Lin, Brian Henderson-Sellers and Igor Hławryszkiewicz	

Software Development: Planning x Agility

Enrico de Sousa Visconti
Escola Politécnica da USP/Microsiga
enrico.visconti@poli.usp.br

Prof. Dr. Edison Spina
Escola Politécnica da USP
edison.spina@poli.usp.br

Abstract: *This paper discusses some of the most common doubts about how much to spend in planning comparing methods from CMM to XP. Best practices are presented and evaluated for each one in the light of the final product characteristics.*

Keywords: *software development, CMM, XP, productivity, reliability*

1. Introduction

Software is an abstract entity that doesn't have physical nor electrical properties, like weight, volume, color or voltage. Software Engineering is, then, different from other engineering areas, it can not use the foundation of natural laws, it has to use only human conceptions [1].

The creation of a suitable mathematical model is an impossible mission. The software development process gets to an uncountable number of different proposals and enhancement discussions targeting stakeholders' satisfaction: customer needs product quality, developers better work conditions, sponsors, profit maximization.

The methodology definition is not a simple matter; there are too many variables in this process, like enterprise differences (size or profile), product application, stakeholders' stile, customer/enterprise relationship, contracts, certifications, etc. The great diversity found takes to an impossible definition of a perfect method as an absolute true.

Too many efforts are being used in this quest. Nowadays, economic globalization, harder competition and customer greater requirements to the "software industries" are growing. The customer stands for higher quality products, lower costs and better support. Customer wants post selling up grades and corrections, companies needs to maximize their investment by increasing efficiency to their processes, cutting costs and getting better productivity.

In chapter 2 one will find some historical data about software development; chapter 3 discuss Software Reliability concepts; chapter 4 presents some important concepts of Software Productivity; chapters 5 and 6 discusses quickness x planning in CMM and XP and chapter 7 presents some conclusions on it.

2. Ascension and Crisis of Software Engineering

The first computational systems emerged by the forties, and were based upon hardware and the programmings were made directly at the machine panel's bottoms [1]. The computational business were based on hardware and there were no differences between programmers and operators. What were based on hardware went to be based on software. It was, still, a strict workmanship. By the next decade, with compilers, linkers, assemblers and programming languages, what were based on hardware became based on software relying only on inspiration and

expertise. The software got some independence upon hardware [3].

At the sixties, computational systems got a great leap. Machines became faster each day. And cheaper. Programming languages and operational systems powerful and friendly. This revolution made the application and the complexity of the computational systems grow. The complexity brought an entirely new problem: the software bugs! The customer confidence was no more the same. The so-called human failure had come to the computers and this were named "software crisis"[2].

To fight this crisis, new technology had to permit reliable software design. In 1968, the name Software Engineering was coined [1]. New techniques like structured design, object oriented and formal description were in. Almost all new developments were focused on fight human failures in software process.

A new decade, the seventies, and new technology, the multi-programming systems, another one and the personal computers fantastic growth and the nineties with the vulgarization of the networking and internet, and the software engineering had to deal with all of it [3] [4]. The software systems are growing faster than software engineering, and another crisis, the software engineering crisis. Bigger teams working in bigger projects lead to greater planning complexity.

An utopian objective for software engineering rises to minimize the computing systems paradox: *the system that works for men is created by men's work*. Software made by computers? This has been a visionary future for years. Software engineering automation is, by now, slower than systems complexity rise. Human factor is, right now, the basis for reliability and productivity in software industry.

3. Software Reliability

Foreseeing software reliability is one of the most technological challenges. It is not function of lifetime as, in software, an abstract entity; there is no wear out. Failures are not caused by any external agents, they already were there, in latency, as errors made by programmers before the usage had begun. Software reliability is focused in two directions, at the project itself and at the validation tests phase (if it is considered independent of the project it self).

During the tests phase it is possible to foresee some of the reliability indexes but during the project phase an even more abstract problem may be evaluated: the human error. This is a qualitative analysis using process quality requirement, and there will be a bond to the human reliability!

A fascinating, but complex, problem is foreseeing the human reliability and what components will influence it.

Human error, before being considered a technological or psychological matter, was considered a religious problem by using freewill (the human freedom to be resembling the Creator). Socrates said: "if men knows clearly all the facts surrounding, certainly will take the right decision"[5]. Socrates took men of the focus, blaming surrounding circumstances that took off men vision from the facts, so the knowledge of the surrounding facts got the reliability flag. Human error got to be a knowledge matter.

To get knowledge, men had to get experience, by their own or by transmitted knowledge. The bigger the man experience, the bigger is his reliability. On the other hand, too many information can cause another obstacle to take right decisions, it is important to get only relevant information into account. Getting full knowledge, on all circumstances, men could simply choose what information is relevant to optimize the solution. Error free is such an intangible

situation for men but, error minimization could be good enough.

A prospective vision put men as a random error source in a reliability model [6]. A retrospective vision studies each form of error and takes actions that makes they never happen or have its effects minimized.

Human error taxonomy took place to a preventive situation. Some of the taxonomies:

a) *Ability, rule and formal knowledge* - Rasmussen [7] - depending upon the basis to decision making. Important for identifying the root cause and permitting a corrective action.

b) *Intuitive and analytical cognition* - Hammond [8] - Studies a cognition scale for decision make process that leads to the error. Offers a tool for evaluation of which process has to be better documented;

c) *Discrete action* - Swain e Guttman [9] - Highly behavioral, analyses only action taken categorized by timing, commission, sequence and oversight.

The knowledge of human error process is the key to reliability through process re-engineering.

4. Software Productivity

Software doesn't take any raw material; it takes only labor to be built. People costs not only by their labor, but also by the training and motivation to good work. All these costs have to be compensated in the product. Neither considering using cheaper people nor bad installations, the only alternative is productivity.

Larry Moore [2] stands that the constant volume of a cube (Fig. 1) may represent software production: the product of budget by time by productivity. A bad use of resources takes to the software higher prices. Not so trained people and/or not enough time leads to re-work, which is the worst

problem of all. Maintenance is 70% of all costs [10] but it is a euphemism. A lot of project is made at this phase. Adaptative or evolutive maintenance are re-work, they implement functionalities that were undertaken during the project. Even at the validation test phase (40% of costs) is to be considered as a part of project test and re-work.

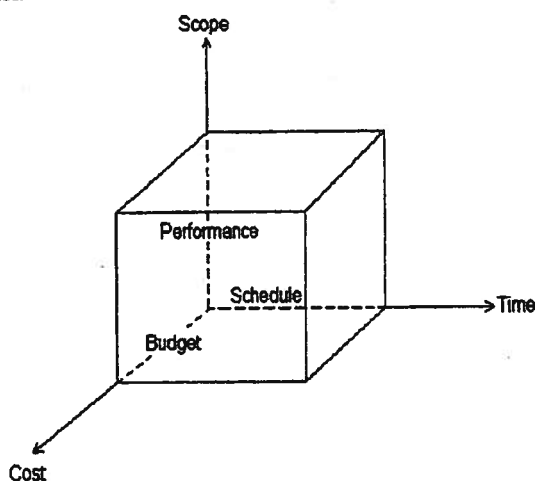


Fig.1 – Software Development Constraints

5. Planning x Quickness

The CMM (Capability Maturity Model) [11] is an example of the process managing methods from TQM (Total Quality Management). Here, the product quality is a function of the managing efforts in the process. The main CMM premise is focused in the process. Better knowledge of the processes takes to better products. DoD (US - Department of Defense) has reports great earnings from this method in software projects. From [12]: 73% lower development costs; 96% lower re-work costs; 37% lower delivery time; 80% less product defect; 21:1 ROI (Return Of Investment).

New studies, in more than 200 software projects, QSM Associates Michael Math verified that almost half of the projects had their initial requirements altered [13].

Client's wishes had more importance than formal requirements. This is a tendency that not may be ignored: rapid changes!

"We needed to know what to build rather than how we should build it" [14]. These ideas lead to the "Agile Methodology" as an alternative to the exigent and complicated consumer market. It is based on the principles [15]:

- Low information exchange cost and quick decision-making;
- Conversation (and sociability!) stead of documentation;
- The customer is a team part, what makes his desires, step by step, clearer;

It is notable that the key point of this "new" process is the human factor [15]. "Software development is a form of art" being directly affected by human capacity of developing new solutions. Based on these principles, Kent Beck has created the concept known as "Extreme Programming" (XP) in 1999. XP was motivated basically by the e-business development [16] and has become the most popular model of the "Agile Metodology".

6. Comparative Visions

These new ideas create some agitation among the software community: traditionalists against revolutionaries. Some responses to the "Agile Manifest":

"Companies focused in innovation instead of production excellence and increasing consistence, previsibility and reliability, that could be gotten without lack of efficiency" [17].

As Steven Rakitin stated, terms let at second importance are the essence, and that ones let to the left are mere excuses to support hackers irresponsible coding, without any discipline from Software Engineering [13].

Aside radicalisms, critics from revolutionary team must be listened, as problems stated are real. Care must be taken saying that XP is enough to the company success. A well-defined and restricted scope exists where these practices could be the only form of organization to the project. Some of these characteristics, that must exist, are listed below:

- *Small team* – even considering that exists some successful records on 250 people teams [15], the team size shall be between 15 and 20 people;
- *Small projects* – less than 250K lines;
- *A propitious ambient to challenges and changes*: these are things not imputable; there would be too many resistance;
- *Customer as part of the project team*: bringing the customer to be part of the solution;
- *Documentation lack*: if it were not important for your client, it would be for you, from were one could get the lessons learned? If there are nothing to be learned, one can let this aside;
- *Flexible time*: any project may have some time variations; the customer presence in the project may bring problems to other projects;
- *Individual talents*: there are no others Kent Becks available in the world to lead teams, too few people has his abilities to put people together and produce a better project than that they would do alone. [18]

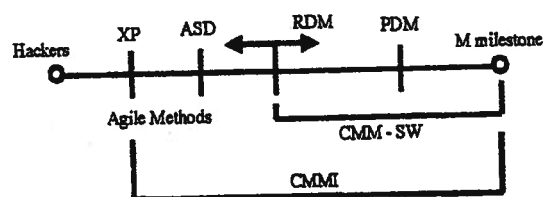
The kind of people addressed here is essential to the successes of the model presented. Without this there would exist only a group generating erratic code. With low-grade people, the problem would be even worst. A disaster! With high-grade professionals, in this model, the company would give no additional value to the project [15].

The CMM and XP models are, in some degree, complimentary models. CMM suggests good practices in key areas to get to excellence without saying how to do. A lot of practices suggested by XP may be used to get there [17]; mainly on requirements changes control as the customer would be part of the development team.

Two extremely defined sides of the spectrum are clear [18]:

- *Hackers*: no discipline, based only upon individual capacity
- *Micromilestone*: inflexible processes based on well-defined contracts, sometimes immutable.

Both sides have problems and virtues. The equilibrium on the project definition is the most important decision to make.



XP – Extreme Programming
 ASD – Adaptive Software Development
 RDM – Risk-Driven Models
 PDM – Plan-Driven Models

Fig.2 – Software development methodologies spectrum

With this problem in mind, DoD-United States Department of Defense - decided to move efforts to CMMI - Capability Maturity Model Integration, publicized it in March 2002 integrating the various earlier CMM models. It is clear, and shown in the fig. 2 how “Agile Methods” were integrated in the new model. It made it more generalist and flexible in its application field.

7. Conclusions

As it is impossible to get a software development model immune to all human errors, these models became the most important investment in software reliability and productivity. A good interpretation of the plausible sources of human errors has to be used to lead the development in this scenario.

The most important critics to the XP model are focused in its name, as “extreme” takes to extreme measures, not always adequate and always restricted domain. The greatest value of the proposed model seems to be the strong inclusion of the customer in the project.

The discussion on “Agile Methods” x traditional models for software development methodology, that, some times, takes almost religious connotation, is an improvement factor to both of them. Criticism by the revolutionaries are the best chance to get some liveliness in the “old” methodology it self, mainly in some bureaucratic ambient like those in CMM.

CMMI contemplates this and help the way to a more productive result. Everybody has a wonderful opportunity to help finding a better way to the software development era.

8. Bibliography

- [1] Pankaj Jalote - "An Integrated Approach to Software Engineering" - New York Ed. Springer-Verlag – 1991
- [2] Larry Moore - "Managing Software Development to Cost and Schedule" – IEEE – 1993 – pp. 475-482
- [3] Abraham Silberchatz et al. - "Operating Systems Concepts" - Addison-Wesley Publishing Company, 3 ed. – 1991

- [4] Andrew S. Tanenbaum - "Redes de Computadores" - Rio de Janeiro - Editora Campus - 1994
- [5] Jose M. Nieves e Andrew P. Sage - "Human and Organizational Error as a Basis for Process Reengineering: With Applications to Systems Integration Planning and Marketing" - IEEE Trans. Syst., Man, Cybern, vol. 28, no. 6, pp. 742-762, 1998
- [6] W. Rouse e S. Rouse - "Analysis and Classification of Human Error" - IEEE Trans. Syst., Man, Cybern, vol. 13, no. 4, pp. 539-549, 1983
- [7] J. Rasmussen, "Skill, rules, knowledge: Signal, signs and symbols; and others distinctions in human performance models" - IEEE Trans. Syst., Man, Cybern, vol. 13, no. 3, pp. 257-267, 1983
- [8] K. R. Hammond, "Intuitive and Analytical Cognition: Information Models" in Concise Encyclopedia of Information Processing in Systems and Organizations, A. P. Sage, Ed. Oxford, U.K.: Pergamon, 1990, pp. 306-312
- [9] A. Swain and H. Guttman, "Handbook of Human Reliability Analysis with Emphasis on Nuclear Power Plant Applications", (NUREG/CR-1278), Washington, DC: Nuclear Regulatory Commission, 1983
- [10] Bertrand Meyer - "Object-Oriented Software Construction" - Ed. Prentice-Hall, 1988
- [11] - Addison Wesley - "The Capability Maturity Model: guides for improving the software process" - Software Engineering Institute, 1997
- [12] - DACS - "A Business Case for SPI Revised - Measuring ROI from Software Engineering and Management", Sept - 1999 <http://www.dacs.dtic.mil/techs/reispi2>
- [13] - Jim Highsmith e Alistar Cockburn - "Agile Software Development: The Business of Innovation", Software Management - Computer, Sept. 2001, pp.120-122
- [14] - Niloy Banerjee e Sukrit Bhattacharya - "Creating an Agile Software Development Organization: A key Factor for Survival in Today's Economy", IEEE, 2002, pp.230-233
- [15] - Jim Highsmith e Alistar Cockburn - "Agile Software Development: The People Factor", Software Management - Computer, Nov. 2001, pp.131-133
- [16] - Arthur English - "Extreme Programming: It's Worth a Look", IEEE - IT Pro, May/June 2002, pp.48-50
- [17] - Mark. C. Paulk - "Extreme Programming from a CMM Perspective", IEEE - Software, Nov/Dec 2001, pp.1-8
- [18] - Barry Bohem - "Get Ready for Agile Methods, with Care", IEEE - Software, Jan 2002, pp.64-69