

Panel: Collaborative Software Engineering – New and Emerging Trends

David Redmiles

Department of
Informatics
University of
California, Irvine
Irvine, CA 92697
+1 949 824 3823

redmiles
@ics.uci.edu

Li-Te Cheng

Collaborative User
Experience Group
T.J. Watson
Research Center
Cambridge, MA
02142
+1 617 577 8500

li-te_cheng
@us.ibm.com

Daniela Damian

Department of
Computer Science
University of
Victoria
Victoria, BC
V8W 3P6
+1 250 721 7292

danielad
@cs.uvic.ca

James Herbsleb

School of Computer
Science
Carnegie Mellon
University
Pittsburgh, PA
15213
+1 412 268 8933

jdh
@cs.cmu.edu

Wendy Kellogg

IBM T.J. Watson
Research Center
P.O. Box 704
Yorktown Heights,
NY 10598
+1 914 784 7826

wkellogg
@us.ibm.com

ABSTRACT

Software engineering has long been a fertile ground for researching collaboration. Early on, software engineers developed tools to support collaboration such as configuration management systems and problem resolution (or bug tracking) systems. These helped support the distributed process that itself became a matter of study. With technological capability and economic incentive, collaborative software engineering became a global phenomenon. Today, social and organizational theories are increasingly applied to improve theoretical understanding and drive further technological development. The goal of this panel is to elucidate new and emerging trends from the technological, empirical, and theoretical perspectives.

Categories and Subject Descriptors

H.5.3 [Group and Organization Interfaces]: Collaborative computing, Computer-supported cooperative work;

General Terms

Design, Management, Human Factors

Keywords

Collaborative Software Engineering, Global Software Engineering, Configuration Management, Open Source Software Development, Activity, Awareness, Software Tools

1. Moderator's Overview

Over the past decades, software engineering settings have provided researchers with insights into many dimensions of human collaboration (e.g. [3]). The domain continues to be a rich source of phenomena and challenges regarding the nature of

collaboration and technology (e.g. [8]). This panel seeks to provide views on emerging trends from technological, empirical, and theoretical perspectives. Some questions that will be addressed are as follows. Why is software engineering a good domain for studying collaboration? What techniques help frame observations or technological experimentation? What are some emerging and exciting observational, theoretical, and technological results? What are some collaboration techniques that succeed today? How much help can technology be in supporting collaboration?

Panelists were selected to provide a wide range of experience and viewpoints. Researchers relatively new to the community are included as well as those with longer-term perspectives.

2. Panelists' Statements

2.1 Li-Te Cheng

Distributed software development is commonplace, programming environments are becoming easier to extend and standardized, and social software is readily gaining acceptance by programmers. Thus, it is an exciting time to consider new studies and tools that address the social side of software engineering [6]. For the tool builder, future "social development environments" might leverage work in the CSCW domain that go beyond just supporting programming environments - such as ubiquitous computing and collaboration technologies emerging on the web. For the researcher, the richness of the new development environments will provide a great context for experiments that could help inform CSCW work outside the software engineering domain. Finally, the nature of today's programmer should be examined - this programmer may have a lot in common with the Blog / MySpace / MMO / Wikipedia generation being studied in CSCW today, and the everyday collaboration tools from that generation will influence software engineering practices in the future.

2.2 Daniela Damian

The rapidly increasing globalization of software industry creates a strong demand to achieve a better understanding of the challenges faced by multi-site software development and to study advanced technologies that successfully support collaborative activities in

global software engineering. Specific areas for technological support are being identified. In particular, requirements management activities are among the most communication-intensive activities in software development and requires effective communication of relevant stakeholders [7]. Traditional problems of requirements engineering are exacerbated in global software teams and an understanding of specific areas for improvement is critical before proposing new requirements engineering processes or techniques for the global software development domain.

2.3 James D. Herbsleb

Software engineering tasks have several unusual properties which, taken together, provide unique opportunities for studying collaboration and designing collaboration technologies [2] [5]. First, the work of individual software designers and developers tends to be highly interdependent. Unlike most forms of design, which are highly constrained by the physical properties of the medium in which the design will be expressed, the varieties of structures and connections is essentially limitless. This relative lack of physical constraints and lack of mature standard decompositions means that software tasks tend to be interdependent in many and complex ways. Thus, it makes a particularly challenging case for supporting collaborative work.

Second, in order to manage the complexity of a software project, software engineers typically use tools that create extremely rich, detailed histories of work on the project. At a minimum, version control systems record every change made by every person to the code, including a detailed record of precisely who, when, and what. Change management systems record every requested change to the system, to who that unit of work was assigned, and all details the workflow of the change until it is completed or abandoned. These records can be used to generate a precise record of the interdependencies of every unit of work, both for research purposes and as the “back end” for tools to support collaboration and awareness.

Finally, as we exploit these detailed records to better understand and support collaboration in software engineering, it also suggests how tools in other kinds of work could be modified to generate similarly detailed histories. Given data that is appropriately stored and linked, one could imagine other tasks being supported by this sort of rich history. For example, think of a lengthy collaboratively-written report in which one could, say, select a graph, pull up the exact version of the spreadsheet from which the data came, trace it back to the database from which the data were taken, look forward to see if any of the data has been updated, find all documents in the company intranet where the data were used, and find the person who contributed each change to every artifact. Such capabilities may ultimately be the big win in the move toward software as a service, where the applications and data all sit on a server, and are delivered over a thin client.

2.4 Wendy Kellogg

A foundational assumption in CSCW is that work is socially organized and cooperative, often in subtle ways that require understanding the context as well as the specific work practice. Along these lines, many documented problems in carrying out cooperative work are about coordinating distributed work, tracking the state of complex projects, discerning the availability of remote colleagues, accommodating the personal quirks or views of a key collaborator, or negotiating conflicting ideas of

what the work is about. Software development has long been recognized as a domain where some of the most difficult problems are beyond technical or simple resource issues, as Brooks' famous 1975 treatise on the Mythical Man-Month attests [1]. Although change tracking systems form a solid (and ubiquitously used) base for collaboration in software engineering, interviews with developers and project managers of large software projects indicate that there is plenty of room for improvement [4]. Our work has explored the use of interactive visualizations of task and social information contained in change tracking systems but which is hard to discern in current systems as a resource for collaboration. The trends of an increasing reliance on global, distributed software engineering, the availability of detailed records from F/LOSS (Free/Libre Open Source Software) development processes, and an increasingly sophisticated user population with respect to collaboration tools and practices makes this domain ever more attractive as a research focus for CSCW.

3. Biographies

David Redmiles is an Associate Professor and Chair of the Department of Informatics in the Donald Bren School of Information and Computer Sciences at the University of California, Irvine. He received his Ph.D. in Computer Science from the University of Colorado, Boulder, in 1992. His dissertation research was in the area of software comprehension and software reuse. Between 1992 and 1994, he worked as part of the research faculty in the Department of Computer Science, at the University of Colorado, Boulder. During this period, he performed research in the area of computer-supported cooperative work and distance learning. Dr. Redmiles joined the faculty at UC Irvine in 1994. There he has researched knowledge-based support for object-oriented design; automated, agent-based support for collecting usage data; tools for supporting awareness; and event-based distributed architectures. He has served on a number of program committees for technical conferences and as a reviewer for NSF. He is the author of a number of technical journal and conference publications. He was Program Co-chair for the 1998 IEEE Conference on Automated Software Engineering was selected to be General Chair of that conference for 2005. He is a joint editor of a 2002 special issue of the Journal of CSCW on Activity Theory and Design. In general, his research interests are in the overlap between software engineering, human-computer interaction, and computer-supported cooperative work.

Li-Te Cheng is a researcher from the Collaborative User Experience (CUE) Group at IBM Research. The CUE group studies and builds social software for the enterprise, information visualization tools, and collaborative development environments. His current research interests lie in intersection of software development, human-computer interaction, and computer supported cooperative work, particularly in extending Eclipse with collaborative capabilities. Li-Te's current work include applying ideas from social tagging to software development (work with Margaret-Anne Storey from the University of Victoria), as well as turning the development environment into an interface for audiences. Before joining IBM, he built and evaluated a vibrotactile feedback system for virtual environments (BSc/MSc at the University of Waterloo) and prototyped a variety of augmented reality systems for wearable computers (PhD at Memorial University of Newfoundland).

Daniela Damian is an Assistant Professor in the Department of Computer Science at the University of Victoria, BC, Canada, where she holds an NSERC University Faculty Award. Her research aims to further the collaborative side of software development practice to support improved software development processes and software products' quality. Her research interests lie in the areas of Software Engineering, Human-Computer Interaction and Computer-Supported Cooperative Work. She leads SEGAL (Software Engineering Global interAction Laboratory). It's mission is to advance the practice of collaborative software engineering, particularly in global software teams which work across geographical, temporal and organizational boundaries. Our research methodologies include learning from advances in research as well as from software developers' work practices.

James D. Herbsleb is an Associate Professor of Computer Science and Director of the Software Industry Center at Carnegie Mellon University. His research interests lie primarily in the intersection of software engineering and computer-supported cooperative work, focusing on such areas as geographically-distributed development teams, open source software development, and more generally on coordination in software engineering. He holds a JD (1980), and a PhD (1984) in psychology from the University of Nebraska, and a MS (1991) in computer science from the University of Michigan. After completing a post-doctoral fellowship at the University of Michigan, he moved to Carnegie Mellon's Software Engineering Institute, where he led an effort to empirically validate the CMM for Software. He then joined the Software Production Research Department at Lucent Technologies, where he initiated and led the Bell Labs Collaboratory Project, which conducted empirical studies and designed collaborative technologies and practices for global software development. He is currently PI on two NSF-funded projects investigating various aspects of collaborative software engineering. He served as co-chair of CSCW 2004.

Wendy A. Kellogg is Manager of Social Computing at IBM's T. J. Watson Research Center. Her work involves designing and studying computer-mediated communication (CMC) systems in groups and organizations. Wendy's work in human-computer interaction (HCI) over the last two decades has spanned theory, evaluation methods, design, and development. She holds a Ph.D. in Cognitive Psychology from the University of Oregon and is author of papers in the fields of HCI, CSCW, and social computing. Wendy served as Technical Papers Co-Chair for CHI

2005, as Technical Program Co-Chair for DIS 2000 conference, and as General Co-Chair for CSCW 2000 and CHI'94. She is a former member of the National Academies of Science Computer Science and Telecommunications Board (CSTB) and was elected ACM Fellow in 2002.

4. REFERENCES

- [1] Brooks Jr., F. P., *The Mythical Man-Month*. 20th Anniversary Edition ed. Addison-Wesley, Reading MA, 1995.
- [2] Cataldo, M., Wagstrom, P., Herbsleb, J.D., Carley, K., *Identification of Coordination Requirements: Implications for the design of collaboration and awareness tools*, Proceedings of CSCW 2006, November 2006, in press.
- [3] Curtis, B., Krasner, H., & Iscoe, N., *A Field Study of the Software Design Process for Large Systems*, Communications of the ACM, V. 31, N. 11, 1988, pp. 1268-1287.
- [4] Halverson, C., Ellis, J., Danis, C., and Kellogg, W.A., *Designing Task Visualizations to Support Coordination of Work in Software Development*, Proceedings of CSCW 2006, November 2006, in press.
- [5] Herbsleb, J.D. & Mockus, A., *An Empirical Study of Speed and Communication in Globally-Distributed Software Development*, IEEE Transactions on Software Engineering, V. 29, N. 3, 2003, pp. 1-14.
- [6] Hupfer, S., Cheng, L., Ross, S., Patterson, J. *Introducing Collaboration into an Application Development Environment*,. Proceedings of CSCW 2004, November 2004, Chicago, IL, pp. 21-24.
- [7] Layman, L., L. Williams, D. Damian, and H. Bures, *Essential communication practices for extreme programming in a global software development team*, Journal of Software and Technology, 2005.
- [8] de Souza, C. R. B., Redmiles, D., Cheng, L.-T., Millen, D., Patterson, J. *Sometimes You Need to See Through Walls—A Field Study of Application Programming Interfaces*, Proceedings of CSCW 2004, November 2004, pp. 63-71.