Software Development Practices in Open Software Development Communities: A Comparative Case Study

(Position Paper)

Walt Scacchi
Institute for Software Research
University of California, Irvine
Irvine, CA 92697-3425 USA
http://www.ics.uci.edu/~wscacchi
wscacchi@ics.uci.edu

Overview

This study presents an initial set of findings from an empirical study of social processes, technical system configurations, organizational contexts, and interrelationships that give rise to open software. "Open software", or more narrowly, open source software, represents an approach for communities of like-minded participants to develop software system representations that are intended to be shared freely, rather than offered as closed commercial products. While there is a growing popular literature attesting to open software [DiBona, Ockman, Stone 1999, Fogel 1999], there are very few systematic studies [e.g., Feller and Fitzgerald 2000, Mockus, Fielding, Herbsleb 2000] that informs how these communities produce software. Similarly, little is known about how people in these communities coordinate software development across different settings, or about what software processes, work practices, and organizational contexts are necessary to their success. To the extent that academic research communities and commercial enterprises seek the supposed efficacy of open software [Smarr and Graham 2000], they will need grounded models of the processes and practices of open software development to allow effective investment of their resources. This study investigates four communities engaged in open software development. Case study methods are used to compare practices across communities.

Understanding open software development practices

Our interest is in understanding the practices and processes of open software development in different communities. We assume there is no prior model or globally accepted framework that defines how open software is or should be developed. So our starting point is to investigate open software practices in different communities to be able to identify what communities members believe are their best practices.

We have chosen four different communities to study. These are those centered about the development of software for:

- *Networked computer game worlds*--first person shooters (e.g., Quake Arena, Unreal Tournament), massively multiplayer online role-playing games (MMORPG, e.g., Everquest, Ultima Online), and others (e.g., The Sims (maxis.com), Neverwinter Nights (bioware.com))

- *Internet infrastructure*--e.g., Apache web server (www.apache.org), InterNet News, Mozilla Web browser, etc.
- *X-ray astronomy and deep space imaging*--e.g., Chandra X-Ray Observatory (http://asc.harvard.edu/swapmeet_top.html) and the European Space Agency's XMM-Newton Observatory (http://xmm.vilspa.esa.es/).
- *Ssoftware systems design* (e.g., ArgoUML software design community now appearing under the banner, argouml.tigris.org).

These communities are constituted by people who identify themselves with the development of one of the four kinds of software just noted. Participants within these communities often participate in different *roles* and contribute *software representations or content* (programs, artifacts, execution scripts, code reviews, comments, etc.) to *Web sites* within each community. Administrators of these cites then serve as gatekeepers in the choices they make for what information to post, when and where within the site to post it, and whether to create a *site map* that constitutes a classification of *site and community domain content*. These people also engage in *online discussion forums* or *threaded email messages* as a regular way to both observe and contribute to discussions of topics of interest to community participants. Finally, in each of the four communities we are examining, participants choose on occasion to author and publish *technical reports or scholarly research papers* about their software development efforts, which are publicly available for subsequent examination and review. Each of these highlighted items point to the public availability of data that can be collected, analyzed, and re-represented within narrative ethnographies or computational process models (Curtis, Kellner, and Over 1992, Kling and Scacchi 1982, Mi and Scacchi 1990, Scacchi 1998,1999). Significant examples of each kind of data can be readily provided for presentation at the Workshop and in the full paper.

Comparative case study framework

The software development practices of the four communities we chose to examine can be compared and contrasted in a number of ways. In this regard, we are conducting case studies in each community. Observations and findings from each such case study can be studied, analyzed, and compared:

- As individual cases
- Within a community
- Multiple cases across two communities
- Multiple cases across all communities

These set of choices for comparison implies that a we can analyze and contrast open software development practices using as many as four different levels of analysis. This multi-level comparative analysis provides a framework for constructing models of practice/process that are both empirically grounded and increasingly general in their scope (Scacchi 1998, 1999). Thus, the comparative case study framework provides a logic that draws on the strength of capturing qualitative data that can provide a rich context for interpretation of case study data though with limited generalization. At the same time, this framework mitigates against the limits of generality assigned to an individual case study through the comparative, crosscutting, and interrelated (e.g.,

hyperlinked) analyses of multiple cases. As before, examples of each level of case data analysis can be readily provided for presentation at the Workshop and in the full paper.

Comparative case studies are important in that they can serve as foundation for the formalization of our findings and process models as a *process meta-model* (Mi and Scacchi 1996). A process meta-model is also used to configure, generate, or instantiate Web-based process modeling, prototyping, and enactment environments that enable modeled processes to be globally deployed and computationally supported (Scacchi and Noll 1997, Noll and Scacchi 1999). Subsequently, we now turn to highlight the software development processes that have been put into practice within different open software communities.

Open software development processes

In contrast to the world of academic software engineering, open software development communities do not seem to readily adopt or practice modern software engineering processes. These communities do develop software that is extremely valuable, generally reliable, widely available, and readily useable within its associated user community. So, what development processes are being routinely used and practiced?

From our studies to date, we have found five types of software development processes being employed across all four communities that merit close examination.

- Requirements analysis and specification
- Coordinated version control, system build, and staged incremental release
- Maintenance as evolutionary redevelopment, refinement, and redistribution
- Project management
- Software technology transfer

Each process will be shown to differ from traditional software engineering prescriptions, though none should be construed as independent or more important than the others should. Furthermore, it appears that these processes may occur concurrent to one another, rather than strictly or partially ordered within a traditional life cycle model. As before, examples of these processes (i.e., process instances) can be provided for presentation at the Workshop and in the full paper.

Conclusions

Open software development practices are giving rise to a new view of how complex software systems can be constructed, deployed, and evolved. Open software development does not adhere to the traditional engineering rationality found in the legacy of software engineering life cycle models or prescriptive standards. Open software development is inherently a complex web of socio-technical processes, development situations, and dynamically emerging development contexts (Kling and Scacchi 1982). This position paper presents an empirical framework that begins to outline some of the contours and dynamics that characterize how open software systems and their associated communities of practice are intertwined and mutually situated to the benefit of both.

Acknowledgements

The research described in this report is supported by a grant from the National Science Foundation #IIS-0083075 for "Understanding Open Software Communities, Processes and Practices", and from the Defense Acquisition University by grant N487650-27803. No endorsement implied. Mark Ackerman and Jack Muramatsu at the UCI Institute for Software Research are collaborators on this research described in this paper.

References

B. Curtis, M.I. Kellner and J. Over, Process modeling, *Communications ACM* 35, 9, 75 - 90, 1992.

C. DiBona, S. Ockman and M. Stone, *Open Sources: Voices from the Open Source Revolution*, O'Reilly Press, Sebastopol, CA, 1999.

J. Feller and B. Fitzgerald, A Framework Analysis of the Open Source Software Development Paradigm, *Proc. 21st. Intern. Conf. Information Systems* (ICIS), 58-69, 2000.

K. Fogel, *Open Source Development with CVS*, Coriolis Press, Scottsdale, AZ, 1999.

R. Kling and W. Scacchi, The Web of Computing: Computer technology as social organization. In M. Yovits (ed.), *Advances in Computers*, Vol. 21, 3-90. Academic Press, New York, 1982.

P. Mi and W. Scacchi, A Knowledge-based Environment for Modeling and Simulating Software Engineering Processes. *IEEE Trans. Knowledge and Data Engineering*, 2(3), 283-294, Sept 1990.

P. Mi and W. Scacchi, A Meta-Model for Formulating Knowledge-Based Models of Software Development. *Decision Support Systems*, 17(3), 313-330. 1996.

A. Mockus, R.T. Fielding, and J. Herbsleb, A Case Study of Open Software Development: The Apache Server, *Proc. 22nd. International Conf. Software Engineering*, Limerick, IR, 263-272, 2000.

J. Noll and W. Scacchi, Supporting Software Development in Virtual Enterprises. *J. Digital Information*, 1(4), February 1999.

W. Scacchi, Modeling, Simulating, and Enacting Complex Organizational Processes: A Life Cycle Approach, in M. Prietula, K. Carley, and L. Gasser (eds.), *Simulating Organizations: Computational Models of Institutions and Groups*, AAAI Press/MIT Press, Menlo Park, CA, 153-168, 1998.

W. Scacchi, Experience with Software Process Simulation and Modeling, *J. Systems and Software*, 46,183-192, 1999.

W. Scacchi and J. Noll, Process-Driven Intranets: Life-Cycle Support for Process Reengineering. *IEEE Internet Computing*, 1(5), 42-49, September-October 1997.

L. Smarr and S. Graham, Recommendations of the Panel on Open Source Software for High End Computing, Presidential Information Technology Advisory Committee (PITAC), http://www.ccic.gov/ac/pres-oss-11sep00.pdf, September 2000.