SOFTWARE PROCESS IMPROVEMENT AND PRACTICE
Softw. Process Improve. Pract. 2006; **11**: 000–000
Published online in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/spip.255

SPIP255

Guest Editorial

Understanding Free/Open Source Software Development Processes•

Walt Scacchi[1], Joseph Feller[2], Brian Fitzgerald[3,*,†],
Scott Hissam[4] and Karim Lakhani[5]
[1] *Institute for Software Research, University of California, Irvine, USA*
[2] *Business Information Systems, University College Cork, Ireland*
[3] *Computer Science and Information Systems, Limerick University, Ireland*
[4] *Software Engineering Institute, Carnegie-Mellon University, USA*
[5] *Sloan School of Management, Massachusetts Institute of Technology, USA*

This article introduces a special issue of *Software Process – Improvement and Practice* focusing on processes found in free or open source software development (F/OSSD) projects. It seeks to provide a background review of research in this area through a review of selected empirical studies of F/OSSD processes. The results and findings from a survey of empirical studies of F/OSSD give rise to an interesting variety of opportunities and challenges for understanding these processes, which are identified along the way. Overall, what becomes clear is that studies of F/OSSD processes reveal a more diverse set of different types of processes than have typically been examined in conventional software development projects. The articles in this special issue further advance understanding of what processes characterize and shape F/OSSD. Copyright © 2006 John Wiley & Sons, Ltd.

KEY WORDS: open source software development; free software development; free/open source software processes

1. INTRODUCTION

This article explores patterns and processes that emerge in free/open source software development (F/OSSD) projects. F/OSSD is a relatively new way of building and deploying large software systems on a global basis, and differs in many interesting ways from the principles and practices traditionally advocated for software engineering (SE) (Feller *et al.* 2005). Hundreds of F/OSS systems are now in widespread use by thousands, or in some cases, millions of end-users. And some of these F/OSS systems (e.g. Mozilla Web browser, OpenOffice productivity suite, Eclipse and NetBeans interactive development environments, KDE and GNOME user interface packages, and most Linux distributions) entail millions of lines of source code. So what is going on here, and how do emerging F/OSSD processes build and sustain these different projects? How might new studies of these processes be used to explore what is new or different?

One of the more significant features of F/OSSD is the formation and enactment of complex software

* Correspondence to: Brian Fitzgerald, Computer Science and Information Systems, Limerick University, Ireland
† E-mail: bf@ul.ie

development processes performed by loosely coordinated software developers and contributors who may be globally dispersed. On some large F/OSSD projects, companies are assigning and paying software development staff to work on F/OSSD projects as part of their job. In contrast, many other developers volunteer their time and skill to such effort, and may only work at their personal discretion rather than as assigned and scheduled. Further, these developers generally provide their own computing resources, and bring their own software development tools with them. However, most F/OSS developers work on software projects that do not typically have a corporate owner or management staff to organize, direct, monitor, and improve the software development processes being put into practice. But how are successful F/OSSD projects and processes possible without regularly employed and scheduled software development staff, or without an explicit regime for software engineering project management? Why will software developers participate in F/OSSD projects? Why and how are large F/OSSD projects sustained? How are large F/OSSD projects coordinated, controlled or managed without a traditional project management team? What is it about the communications, roles, and artifacts that enable some projects to persist but not others? Why and how might these answers to these questions change over time? These are the kinds of questions raised in this article, and in the articles that follow in this Special Issue of *Software Process – Improvement and Practice*.

The remainder of this article is organized as follows. The next section provides further background on what F/OSSD is and what is already known about F/OSSD practices, based on both trade studies and systematic empirical studies. This survey focuses attention on identifying opportunities and challenges in understanding F/OSSD processes through empirical studies. Following this are brief overviews of each of the five articles that were selected for this Special Issue after a comprehensive submission, review, and selection effort. A final discussion then argues why the software process research community may itself want to adopt F/OSSD practices for sharing and enabling others to modify and share explicit descriptions, representations, models, and related 'software process source code' within and across the software, information systems, computer science, and social science research communities.

1.1. What is Free/Open Source Software Development?

Free (as in freedom) software and open source software (OSS) are often labeled or treated as the same thing. However, there are important differences between them with regards to the licenses assigned to the respective software, and to the beliefs/ideology of their practitioners of how and why software should be developed for sharing, modification, reuse, and redistribution. Free software generally appears licensed with the GNU general public license (GPL), while OSS may use either the GPL or some other license that allows for the integration of software that may not be free software. Free software is a social movement (Elliott and Scacchi 2004), whereasopen source software development (OSSD) is a software development methodology, according to free software advocates like Richard Stallman and the Free Software Foundation (Gay 2002). However, free software is always available as OSS, but OSS is not always free software[1]. This is why it is often appropriate to refer to F/OSS or FLOSS (L for *Libre*, where the alternative term of 'libre software' has popularity in some parts of the world) in order to accommodate similar and often indistinguishable approaches to software development. Subsequently, for the purposes of this article, focus is directed at F/OSSD processes, rather than to software licenses and social movements associated with free or OSS, though each may impinge on F/OSSD processes.

F/OSSD is mostly not about SE, at least not as SE is portrayed in modern SE textbooks. Similarly, F/OSSD is not SE done poorly. Instead, F/OSSD is a different, somewhat orthogonal approach to the development of software systems where much of the development activity is openly visible, development artifacts are publicly available over the Web, and generally there is no formal project management regime, budget or schedule. F/OSSD is oriented towards the joint development of a community of developers and users concomitant with the software system of interest.

F/OSS developers have typically been end-users of the F/OSS they develop, although this

[1] Thus, at times it may be appropriate to distinguish conditions or events that are generally associated or specific to either free software development or OSSD, but not both.

appears to be changing somewhat. Similarly, many end-users often participate in and contribute to F/OSSD efforts by providing feedback, bug reports, and usability concerns. There is also widespread recognition that F/OSSD projects can produce high quality and sustainable software systems that can be used by thousands to millions of end-users (Mockus *et al*. 2002).

Subsequently, it is reasonable to assume that F/OSSD processes are not necessarily of the same type, kind, or form found in SE projects that follow the processes described in modern SE textbooks. While such approaches might be used within an SE project, there is no basis found in the principles of SE laid out in textbooks that would suggest SE projects typically adopt or should practice F/OSSD methods. Subsequently, what is known about SE processes, a development organization's process capability, and how to improve its development processes may not be equally applicable to F/OSSD processes, without some explicit rationale or empirical justification. Thus, it is appropriate to review some of what is known so far about F/OSSD.

2. RESULTS FROM RECENT STUDIES OF F/OSSD

There are studies that offer some insight or findings on F/OSSD practices where each, in turn, reflects on different kinds of processes that are not well understood at this time. These are systematic empirical studies of F/OSSD projects using small/large research samples and analytical methods drawn from different academic disciplines. Both kinds of studies stand in contrast to the popular examination of F/OSSD practices offered by F/OSS advocates (e.g. DiBona *et al*. 1999, Pavelicek 2000, Raymond 2001). These popular treatments tend to be grounded in personal experiences of the authors, rather than through careful systematic study, though such experiences are valuable because they are often a source of insight or questions for further inquiry.

A number of Web-based repositories of research articles that report on studies on F/OSSD projects have begun to appear. Among them are those at MIT (opensource.mit.edu) with over 200 articles contributed, and at University College Cork in Ireland (opensource.ucc.ie), which features links or citations to multiple special issue journals focusing on F/OSSD (e.g. *Information Systems Journal*, 11(4) and 12(1), 2001–2002, *IEE Proceedings – Software*, 149(1), 2002, *Research Policy*, 32(7), 2003, and *IEEE Software*, 21(1), 2004), and to proceedings from international workshops of OSS research (e.g. 1st through 5th Workshops on Open Source Software Engineering, held in conjunction with the International Conferences on Software Engineering, 2001–2005). Rather than attempt to survey the complete universe of studies in these collections, since the majority of these studies do not address software processes, the choice instead is to examine a set of studies[2] that raise interesting issues or challenging problems for software process research and practice.

One important qualifier to recognize is that the studies below examined carefully selected F/OSSD projects, or a sample of projects, so the results presented should not be assumed to apply to all F/OSSD projects, or to projects that have not been studied. Furthermore, it is important to recognize that F/OSSD is no silver bullet that resolves the software crisis. Instead it is fair to recognize that most of the nearly 100 000 F/OSSD projects associated with Web portals like SourceForce.org have very small teams of two or less developers (Madey *et al*. 2004), and most projects are inactive or have yet to release any operational software. However, there are now at least a few thousand F/OSSD projects that are viable and ongoing, so that there is a sufficient universe of diverse F/OSSD projects to investigate, understand, as well as to model and simulate their software processes. Consequently, consider the research findings reported or studies cited below as starting points for further investigation, rather than as defining characteristics of most or all F/OSSD projects or processes.

2.1. Comparing F/OSSD and SE Processes

The first category of related studies seek to identify and compare software development processes found in F/OSSD projects with those described or prescribed for SE projects, rather than just the resulting software products (Paulson *et al*. 2004). Mockus *et al*. (2002), in one of the most cited

[2] The articles included here address some software development process, such as the OSS design process, in the body of their article (as found using search engines), or address topics that heretofore have not appeared in prior software process studies.

studies of F/OSSD, briefly describe the processes accounting for the development of the Apache Web Server and the Mozilla Web Browser. However, such an account does not provide sufficient content to directly compare them to traditional SE processes. In contrast, Reis and Fortes (2002) provide one of the first in-depth examinations of the overall process accounting for the development of the Mozilla Web Browser. They identify different developer roles, tools being used, artifacts created, and activities performed, which potentially provides adequate information for modeling and comparing the process.

Scacchi (2002) provides a narrative description of the software requirements process found in a sample of F/OSSD projects and compares it to the requirements engineering process portrayed in modern SE textbooks. The F/OSSD projects examined span applications in application domains including Internet infrastructure, networked computer games, astrophysics, and academic software design research. In a related study (Scacchi 2004), he identifies differences in software processes for requirements and design, configuration management, evolution, project management, and software technology transfer from those in SE texts as found in the comparative study of multiple F/OSSD projects of a common type and networked computer games. Further, in two additional studies, he examines data and models accounting for software evolution (Lehman 2002) compared to those emerging for F/OSSD (Scacchi 2005a), and also emerging sociotechnical processes found in F/OSSD projects that intermingle social (e.g. team, group, and individual) and technical development processes (Scacchi 2005b, Truex *et al*. 1999). All of these studies describe processes found in different F/OSSD projects using narrative descriptions or models. Furthermore, recent work describes how these processes have been modeled and simulated in a variety of notational, computational, and ethnographic schemes (Scacchi *et al*. 2005c).

Finally, other recent efforts to model and simulate F/OSSD processes of different kinds has begun to appear. Antoniades *et al*. (2004) and Smith *et al*. (2004) both provide simulation models of processes accounting for the overall development or evolution of multiple F/OSSD projects. Both efforts rely on models expressed as continuous functions through either algebraic formulae or systems of equations. Such an approach to process simulation and modeling appears well matched to Systems Dynamics-based process simulation tools. In contrast, Jensen and Scacchi (2004, 2005) model and reenact processes found in a small sample of OSSD projects using language-based process models and a process reenactment simulator following techniques developed for analyzing traditional SE processes (Noll and Scacchi 2001, Scacchi 1999). The ability to reenact F/OSSD processes provides an approach for how to independently examine and validate whether the captured process reflects the observed practice, as well as makes the modeled processes available for reuse, tailoring, improvement, or practice in other F/OSSD projects.

2.2. Motivating, Joining, Participating, and Contributing to F/OSSD Projects

One of the most common questions about F/OSSD projects is why software developers will join and participate in such efforts, often without pay for sustained periods of time. Accordingly, we might then ask whether or how the participation of developers affects what gets done in a F/OSSD project in terms of which processes are engaged, as well as understanding how processes for recruiting and integrating new developers into F/OSSD projects operate. A number of surveys of F/OSS developers (Ghosh and Prakash 2000, Hars and Ou 2002, Hann *et al*. 2002, Lakhani *et al*. 2002, Hertel *et al*. 2003) have begun to pose such questions, and the findings reveal the following.

First, F/OSS developers generally find the greatest benefit from participation is the opportunity *to learn and to share what they know* about software system functionality, design, methods, tools, and practices associated with specific projects or community leaders (Lakhani *et al*. 2002). F/OSSD is a venue for learning for individuals, project groups, and organizations, and learning organizations are ones which can continuously improve or adapt their processes and practices (Nakakoji *et al*. 2002, Huntley 2003, Ye and Kishida 2003). However, though much of the development work in F/OSSD projects is unpaid or volunteer, individual F/OSS developers often benefit with higher average wages and better employment opportunities (at present), compared to their peers who lack F/OSSD experience or skill (Hann *et al*. 2002, Lerner and Tirole 2002).

Second, F/OSS developers appear to really enjoy their F/OSSD work (Hertel *et al*. 2003), and to be recognized as trustworthy and reputable contributors (Stewart and Gosain 2001). F/OSS developers also self-select the technical roles they will take on as part of their participation in a project (Ye and Kishida 2003, Gacek and Arief 2004), rather than be assigned to a role in a traditionally managed SE project, where the assigned role may not be to their liking.

Third, many F/OSS developers participate in and contribute to multiple F/OSSD projects. In one study, 5% of developers surveyed reported participating in 10 or more F/OSSD projects (Hars and Ou 2002). However, a small group of core developers who control the architecture and direction of development typically develop the vast majority of F/OSS released by a project. Subsequently, most participants typically contribute to just a single module, though a small minority of modules may include patches or modifications contributed by hundreds of contributors (Ghosh and Prakash 2000).

Consequently, how and why software developers will join, participate in, and contribute to an F/OSSD project seems to represent a new kind of process affecting how F/OSS is developed and maintained (von Krogh *et al*. 2003, Scacchi 2005b). Subsequently, modeling and simulating what this process is, how it operates, and how it affects software development is an open research challenge.

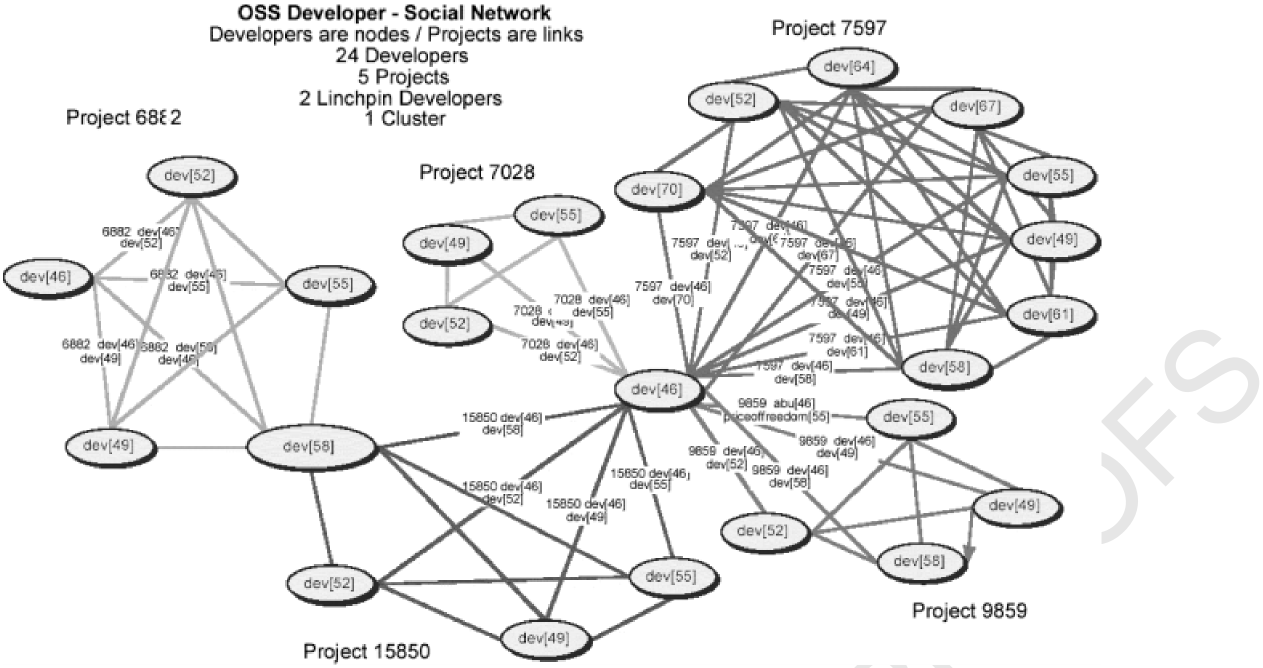
2.3. Alliance Formation, Social Networking, Community Development, and Software Development

How does the gathering of individual F/OSS developers give rise to a more persistent project team or self-sustaining community? Through choices that developers make for their participation and contribution to an F/OSSD project, they find that there are like-minded individuals who also choose to participate and contribute to a project. These software developers find and connect with each other through F/OSSD Web sites and online discourse (e.g. threaded e-mail discussions) (Monge *et al*. 1998), and they find they share many technical competencies, values, and beliefs in common (Crowston and Scozzi 2002, Espinosa *et al*. 2002, Elliott and Scacchi 2004). This manifests itself in the emergence of an occupational network of F/OSS developers (Elliott and Scacchi 2003).

Becoming a central node in a social network of software developers that interconnects multiple F/OSS projects is also a way to accumulate social capital and recognition from peers. However, it also enables the merger of independent F/OSS systems into larger composite ones that gain the critical mass of core developers to grow more substantially and attract ever larger user-developer communities (Madey *et al*. 2004, Scacchi 2005b). 'Linchpin developers' (Madey *et al*. 2004) participate in or span multiple F/OSSD projects. In so doing, they create alliances between otherwise independent F/OSSD projects (Hars and Ou 2002). Figure 1 depicts an example of a social network of 24 F/OSS developers within 5 F/OSS projects that are interconnected through two linchpin developers (Madey *et al*. 2004). Such interconnection enables small F/OSS projects to come together as a larger social network with the critical mass (Marwell and Oliver 1993) needed for their independent systems to be merged and collectively experience more growth in size, functionality, and user base. F/OSSD Web sites also serve as hubs that centralize attention on what is happening with the development of the focal F/OSS system, its status, participants and contributors, discourse on pending/future needs, etc.

Thus interesting problems arise when investigating how best to capture or represent the processes of alliance formation and interproject social networking, and how such processes can be shown to facilitate or constrain F/OSSD activities, tool usage, and preference for which development artifacts are most valued by project participants.

Developing F/OSS systems is a community and project team building process that must be institutionalized within a community (Sharma *et al*. 2002, Smith and Kollock 1999, Preece 2000, Ye *et al*. 2004) for its software informalisms (artifacts) and tools to flourish. Downloading, installing, and using F/OSS systems acquired from other F/OSS Web sites is also part of a community building process (Kim 2000). Adoption and use of F/OSSD project Web sites is a community-wide practice of how to publicize and share F/OSS project assets. These Web sites can be built using F/OSSD Web site content management systems (e.g. PhP-Nuke) to host project contents that can be served using F/OSS Web servers (Apache), database systems (MySQL) or application servers (JBoss), and increasingly accessed via F/OSS



Color Figure - Online only

AQ2

Figure 1. A social network that links 24 developers in five projects through two key developers into a larger F/OSS project community• (Madey *et al*. 2004)

Web browsers (Mozilla and Firefox). Furthermore, ongoing F/OSSD projects may employ dozens of F/OSS development tools, whether as standalone systems like the software version control system CVS, as integrated development environments like NetBeans or Eclipse, or as subsystem components of their own F/OSS application in development. These projects similarly employ asynchronous systems for project communications that are persistent, searchable, traceable, public, and globally accessible (Yamauchi *et al*. 2000).

F/OSS systems, hyperlinked artifacts and tools, and project Web sites serve as venues for socializing, building relationships and trust, sharing and learning with others. Community building, alliance forming, and participatory contributing are essential and recurring activities that enable F/OSSD projects to persist without central corporate authority. Linking people, systems, and projects together through shared artifacts and sustained online discourse enables a sustained sociotechnical community, information infrastructure (Jensen and Scacchi 2005), and a network of alliances (Monge *et al*. 1998) to emerge.

For this reason, problems arise when investigating how best to capture and represent the F/OSSD processes that facilitate and constrain the codevelopment and coevolution of F/OSS project communities and the software systems they produce. The point is not to separate the development and evolution processes of the software system from its community, since each is codependent on the other, and the success of one depends on the success of the other. Thus, they must be captured, understood, modeled, and simulated/reenacted as integrating and intertwining processes.

2.4. Software Evolution in a Multiproject Software Ecosystem

As noted above, many F/OSSD projects have become interdependent through the networking of software developers, development artifacts, common tools, shared Web sites, and computer-mediated communications. What emerges from this is a kind of multiproject software ecosystem (Highsmith 2002), whereby ongoing development and evolution of one F/OSS system gives rise to propagated effects, changes, or vulnerabilities in one or more of the projects linked to it (Jensen and Scacchi 2005). These interdependencies are most apparent when F/OSSD projects share source code modules or components. In such situations, the volume of source code of an individual F/OSSD project may appear to grow at a superlinear or exponential rate (Scacchi 2005a, Schach *et al*. 2002, Smith *et al*. 2004). Such an outcome, which economists and political scientists refer to as a 'network externality' (Ostrom *et al*. 1990), may be due to the import or integration of shared components, or the replication and tailoring of device, platform, or internationalization specific code modules. Such system growth patterns might challenge the well-established laws of software evolution (Lehman 1980, 2002). Thus, software evolution in a multiproject F/OSS ecosystem is a process of coevolution of interrelated and interdependent F/OSSD projects, people, artifacts, tools, code, and project-specific processes.

Software evolution in a multiproject F/OSS ecosystem also suggests attending to social or technological mechanisms that provide some form of 'natural selection'. In biological ecosystems, natural selection provides an account of why some species flourish and adapt in response to environmental pressures, such as shortage of food sources or the rise of new predators, while other species that do not adapt progressively disappear or become extinct. In F/OSS ecosystems, a diversity of software system variants often appear as distinct projects. For example, there are a number of F/OSS operating systems with projects based on variants of the Unix operating system – Linux, FreeBSD, OpenBSD, Darwin, GNU Hurd, etc.–and each of these may have multiple subvariants (or forked distributions) like Debian GNU/Linux, SUSE Linux, Red Hat Linux, and hundreds of others. Web browsers, software build/make tools, database management systems, file management utilities, content management systems, and other types of software systems can be found by the dozens, perhaps reflecting their development in different F/OSS ecosystem niches. Similarly, one can readily find at F/OSS project portals like SourceForge.net, Freshmeat.net or Savannah.gnu.org, multiple projects developing the same type of software system, but with variations in software architecture, choice of functional components, choice of programming language, and project contributors. However, in some software domains, a dominant software system and project has emerged to effectively displace alternative variants by a large majority, like the Apache Web server, though such dominance has not completely eliminated the contending alternative F/OSS project efforts. As such, accounting for such evolutionary adaptation in response to emerging technological opportunities (new tools) or limited access to more established F/OSS projects or core developers is thus a challenge for those seeking to understand the processes of software evolution across a software ecosystem (Lehman 2002, Nakakoji *et al*. 2002, Smith *et al*. 2004, Ye *et al*. 2004).

Last, it seems reasonable to observe that the world of F/OSSD is not the only place where multiproject software ecosystems emerge, as software sharing or reuse within traditional software development enterprises is common (Highsmith 2002, Jensen and Scacchi 2005). However, the process of the coevolution of software ecosystems found in either traditional or F/OSSD projects in mostly unknown. Thus, software coevolution within an F/OSS ecosystem represents an opportunity for research that investigates understanding such a software evolution process through studies supported by techniques for modeling and simulating coevolving processes.

Overall, the sample of F/OSSD research studies and findings presented above reveals a number of interesting challenges for research in understanding F/OSSD processes. However, these studies are all grounded in an empirical basis where different types of processes are being examined in different types of F/OSSD projects of varying sample size and data collection methodology. So the fundamental problem at hand is how to organize, reframe, and make clear what the challenges are in researching, improving, and practicing F/OSSD processes. The articles in this special issue help provide new insights and findings for better understanding the problem and challenges.

3. ARTICLES SELECTED FOR THE SPECIAL ISSUE

Five articles were selected as a result of the submission and review process from a pool of 21 submitted articles for inclusion in this Special Issue. In our first article, 'Evaluation of Free/Open Source Software Products through Project Analysis,' David Cruz, Thomas Wieland, and Alexander Ziegler introduce a systematic approach for supporting a decision to incorporate F/OSS products into a larger context, such as a software or enterprise-wide system. The process of evaluating and integrating

commercially available off-the-shelf software (often referred to as COTS software) has been written and described at length in academic and industrial literature. Often, such evaluations are conducted to avoid unnecessary risks, including the technical (Hissam and Plakosh 1999) and mission needs of the system to which the evaluated software
AQ3 is to be incorporated (•Carney *et al.* 2003). Often, many of the same considerations apply to F/OSS products. However, as this article points out, there are additional and relevant aspects of an F/OSS project that produces the F/OSS product that should be taken into consideration. The authors nicely characterize these considerations into functional, technical, organizational, legal, economical and political aspects and thereby provide a broader perspective on F/OSS products when performing such evaluations.

In 'Information Systems Success in Free and Open Source Software Development: Theory and Measures', Kevin Crowston, Hala Annabi, James Howison, and Chengetai Masango address a key gap in FLOSS research by seeking to define what 'success' means in a FLOSS context. They derived a range of potential measures from the Information Systems (IS) literature, including system and information quality, user satisfaction, user experience, individual and organizational impacts, and then added a number of additional measures specific to the dynamics of the FLOSS development process. The list of measures was then refined, operationalized, and validated through empirical studies based in the SlashDot and SourceForge communities. The article makes a welcome contribution by providing a solid instrument for the future evaluation of FLOSS processes, and one that will mature and increase in its utility with further use.

David Nichols and Michael Twidale report on their study of 'Usability Processes in Open Source Software.' They describe mechanisms, techniques, and technology used in F/OSS projects like Mozilla and GNOME. Both of these projects develop systems that include substantial user interface components and functionality, and so they are primary candidates to consider how F/OSS development processes and practices embrace or ignore usability concerns. They use examples drawn from bug reporting and discussion systems to highlight both the current practice, and how it might be revised to realize higher quality F/OSS development outcomes and easier to use application systems. This in turn may give rise to recognizing how F/OSS projects need to both address their ease of development (or 'developability') and the usability of the resulting software systems. This is particularly true, as Nichols and Twidale observe, when developers and users are geographically dispersed, have limited resources to affect current processes, and may lack easily accessible sources of expertise about the functionality of the systems being developed, as well as how to make such functionality easy to use.

Douglas Schmidt and colleagues at Vanderbilt University and University of Maryland at College Park investigate 'Techniques and Processes for Improving the Quality and Performance of Open Source Software.' Their research complements the work of Crowston *et al.* in this issue, but whereas that article sought to define success, Schmidt *et al.* tackle the more specific question of software quality. They describe some of the challenges associated with FLOSS, and explore the ways in which quality assurance (QA) processes that are specifically designed for FLOSS processes can help address these challenges. They support their work with empirical examinations of FLOSS projects using these QA processes, and conclude with extremely practical findings of direct benefit to FLOSS practitioners.

Katherine Stewart and Sanjay Gosain examine 'The Moderating Role of Development Stage in Affecting Free/Open Source Software Project Performance.' This is an important contribution towards understanding how social factors like team trust and ideology interact with the development process of a project and impact objective and subjective outcomes like task completion, number of developers mobilized, and perceived effectiveness. Using data from 67 F/OSS communities, they show that the dynamics of performance change as a project moves through various development stages. Their results suggest that objective measures of project performance tend to improve over time and with increases in development stage, while subjective assessments depend to a greater extent on the project administrators' experience. For example, the importance of trust in teams varies on the basis of the development stage of the project and the performance criteria. Overall they have presented a sophisticated view of the interaction between the social and technical factors in a F/OSSD project throughout its development process.

4. DISCUSSION

F/OSSD projects represent and offer new publicly available data sources of a size, diversity, and complexity not previously available for research, on a global basis. Software process research and application has traditionally relied on an empirical basis in real-world processes for analysis, validation, or improvement. However, such data has often been scarce, costly to acquire, and is often not available for sharing or independent reanalysis for reasons including confidentiality or nondisclosure agreements. In contrast, F/OSSD projects and project repositories contain process data and product artifacts that can be collected, analyzed, shared, and reanalyzed in a free and open source manner. The articles in this Special Issue draw upon publicly available data and artifacts in their analyses. F/OSSD therefore poses the opportunity to favorably alter the costs and constraints of accessing, analyzing, and sharing software process and product data, metrics, and data collection instruments. F/OSSD is thus poised to alter the calculus of empirical SE, information systems, and perhaps even computer science, while software process research is an arena that can take advantage of such a historically new opportunity.

Finally, one important dimension that has not yet been addressed in this article is whether and how the software process research community might adopt F/OSSD practices themselves. For example, one traditional barrier to engaging students in software process studies is the paucity of free or low-cost modeling and simulation tools. Sharing one's software process models and simulations with colleagues is difficult at present, if they must buy new and unfamiliar tools. The ability to reuse, reanalyze, or extend a colleague's models or simulations is similarly limited. The community needs and should directly benefit from F/OSS process models, tools, and process data/model repositories that can be easily acquired or shared, studied, modified, and redistributed to the mutual benefit of all. Similarly, it can also be noted that it further serves the collective interest of the community to consider how to develop a globally shared and interoperable information infrastructure for data sharing, modeling, and simulating software processes[3]. This is true, whether these processes are found in SE or F/OSSD projects. As a consequence, these are all opportunities for the software process research community to realize and pursue. After all, we are the ones who will benefit from efforts to develop such free (as in freedom) and open source resources, as well as further our collective learning and community building efforts.

[3] Efforts like the FLOSSmole repository (www.flossmole.org) and the SourceForge.net Research Data repository (www.nd.edu/∼oss/Data/data.html) are two emerging examples in this area.

5. CONCLUSIONS

Free and OSS development is emerging as an alternative approach for developing large software systems. New types and new kinds of software processes are emerging within F/OSSD projects, as well as new characteristics for development project success, when compared to those found in traditional industrial software projects and those portrayed in software engineering textbooks. As a result, F/OSSD offers new types and new kinds of processes to research, understand, improve, and practice. Similarly, understanding how F/OSSD processes are similar to or different from SE processes is an area ripe for further research and comparative study. Many new research opportunities exist in the empirical examination, modeling, simulation, improvement, and practice of F/OSSD processes.

Through a survey of empirical studies of F/OSSD projects and other analyses presented in this article, it should be clear there is an exciting variety and diversity of opportunities for new research aimed at understanding and improving the practice of F/OSSD. Thus, you are encouraged to consider how your efforts to research or apply software process concepts, techniques, or tools can be advanced through studies that examine processes found in F/OSSD projects, and that practice free or open source sharing, reuse, and extension of *software process* data, artifacts, models, and public repositories.

ACKNOWLEDGEMENTS

The research described in this report is supported by grants from the US National Science Foundation #0083075, #0205679, #0205724, and #0350754; from Science Foundation Ireland #02/IN.1/I108; and from the EU to the CALIBRE project. No endorsement implied.

REFERENCES

Antoniades IP, Samoladas I, Stamelos I, Angelis L, Bleris GL. 2004. Dynamic Simulation models of the open source development process. In *Free/Open Source Software Development*, Koch S (ed.). Idea Group Publishing: Hershey, PA, 174–202.

Crowston K, Scozzi B. 2002. Open source software projects as virtual organizations: competency rallying for software development. *IEE Proceedings Software* **149**(1): 3–17.

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

Elliott M, Scacchi W. 2003. Free software developers as an occupational community: Resolving conflicts and fostering collaboration. In *Proceedings of the ACM International Conference on Supporting Group Work*, Sanibel Island, FL, November 2003, 21–30.

Elliott M, Scacchi W. 2004. Free software development: Cooperation and conflict in A virtual organizational culture. In *Free/Open Source Software Development*, Koch S (ed.). Idea Group Publishing: Hershey, PA, 152–172.

Espinosa JA, Kraut RE, Slaughter SA, Lerch JF, Herbsleb JD, Mockus A. 2002. Shared mental models, familiarity, and coordination: ∼A multi-method study of distributed software teams. In *International Conference Information Systems*, Barcelona, Spain, December, 2002 425–433.

Feller J, Fitgerald B, Hissam S, Lakhani K. 2005. *Perspectives on Free and Open Source Software*. MIT Press: Cambridge, MA.

Gacek C, Arief B. 2004. The many meanings of open source. *IEEE Software* **21**(1): 34–40.

Gay J (ed.). 2002. *Free Software, Free Society: Essays by Richard M. Stallman*. Free Software Foundation: Cambridge, MA.

Ghosh R, Prakash VV. 2000. The orbiten free software survey. *First Monday* **5**(7): Also see http://www.infonomics.nl/FLOSS/ •for further information. AQ4

Hann I-H, Roberts J, Slaughter S, Fielding R. 2002. Economic incentives for participating in open source software projects. In •*Proceedings of the Twenty-Third International Conference on Information Systems*, December 2002 365–372. AQ5

Hars A, Ou S. 2002. Working for free? Motivations for participating in open source projects. *International Journal of Electronic Commerce* **6**(3): 25–39.

Hertel G, Neidner S, Hermann S. 2003. Motivation of software developers in open source projects: an internet-based survey of contributors to the Linux kernel. *Research Policy* **32**(7): 1159–1177.

Highsmith J. 2002. *Agile Software Development Ecosystems*. Addison-Wesley: Boston, MA.

Hissam S, Plakosh D. 1999. *COTS in the Real World: A Case Study in Risk Discovery and Repair*, CMU/SEI-99-TN-003, Software Engineering Institute, Carnegie Mellon University: Pittsburgh, PA, http://www.sei.cmu.edu/publications/documents/99.reports/99tn003/99tn003abstract.html.

Huntley CL. 2003. Organizational learning in open-source software projects: An analysis of debugging data. *IEEE Transactions on Engineering Management* **50**(4): 485–493.

Jensen C, Scacchi W. 2004. Collaboration, leadership, and conflict negotiation in the NetBeans.org community. In *Proceedings of the 4th Workshop on Open Source Software Engineering*, Edinburgh, UK, May 2004.

Jensen C, Scacchi W. 2005. Process modeling across the web information infrastructure. *Software Process – Improvement and Practice* **10**(3): 255–272.

Kim AJ. 2000. *Community Building on the Web: Secret Strategies for Successful Online Communities*. Peachpit Press: Berkeley, CA.

Lakhani KR, Wolf B, Bates J, DiBona C. 2002. The Boston Consulting Group Hacker Survey, July. http://www.bcg.com/opensource/BCGHackerSurvey-OSCON24July02v073.pdf.

Lehman MM. 1980. Programs, life cycles, and laws of software evolution. *Proceedings of the IEEE* **68**: 1060–1078.

Lehman MM. 2002. Software evolution. In *Encyclopedia of Software Engineering*, 2nd edn, Marciniak J (ed.). John Wiley and Sons: New York, 1507–1513; Also see 2002. Software evolution and software evolution processes *Annals of Software Engineering* **12**: 275–309.

Lerner J, Tirole J. 2002. Some simple economics of open source. *Journal of Industrial Economics* **50**(2): 197–234.

Madey G, Freeh V, Tynan R. 2004. Modeling the F/OSS community: A quantative investigation. In *Free/Open Source Software Development*, Koch S (ed.). Idea Group Publishing: Hershey, PA, 203–221.

Marwell G, Oliver P. 1993. *The Critical Mass in Collective Action: A Micro-Social Theory*. •Cambridge University Press. AQ6

Mockus A, Fielding R, Herbsleb JD. 2002. Two case studies of open source software development: Apache and Mozilla. *ACM Transactions on Software Engineering and Methodology* **11**(3): 309–346.

Monge PR, Fulk J, Kalman ME, Flanagin AJ, Parnassa C, Rumsey S. 1998. Production of collective action in alliance-based interorganizational communication and information systems. *Organization Science* **9**(3): 411–433.

Nakakoji K, Yamamoto Y, Nishinaka Y, Kishida K, Ye Y. 2002. Evolution patterns of open-source software systems and communities. In •*Proceedings of the 2002 International Workshop Principles of Software Evolution*, 76–85.

Noll J, Scacchi W. 2001. Specifying process-oriented hypertext for organizational computing. *Journal of Network and Computer Applications* **24**(1): 39–61.

Ostrom E, Calvert R, Eggertsson T (eds). 1990. *Governing the Commons: The Evolution of Institutions for Collective Action*. •Cambridge University Press.

Paulson JW, Succi G, Eberlein A. 2004. An empirical study of open-source and closed-source software products. *IEEE Transactions On Software Engineering* **30**(4): 246–256.

Pavelicek R. 2000. *Embracing Insanity: Open Source Software Development*. SAMS Publishing: Indianapolis, IN.

Porter AA, •Siy HP, Toman CA, Votta LG. 1997. An experiment to assess the cost-benefits of code inspections in large scale software development. *IEEE Transactions On Software Engineering* **23**: 329–346.

Preece J. 2000. *Online Communities: Designing Usability, Supporting Sociability*. John Wiley and Sons: Chichester, UK.

Raymond ES. 2001. *The Cathedral & The Bazaar: Musings on Linux and Open Source by an Accidental Revolutionary*. O'Reilly Media: Sebastopol, CA.

Reis CR, Fortes RPM. 2002. An overview of the software engineering process and tools in the Mozilla project. In *Proceedings of the Workshop on Open Source Software Development*, Newcastle, UK, February 2002.

Scacchi W. 1999. Experience with software process simulation and modeling. *Journal of Systems and Software* **46**(2/3): 183–192.

Scacchi W. 2002. Understanding the requirements for developing open source software systems. *IEE Proceedings–Software* **149**(1): 24–39.

Scacchi W. 2004. Free/open source software development practices in the computer game community. *IEEE Software* **21**(1): 59–67.

Scacchi W. 2005a. Understanding free/open source software evolution. In *Software Evolution and Feedback*, Madhavji NH, Lehman MM, Ramil JF, Perry D (eds). John Wiley and Sons: New York, to appear.

Scacchi W. 2005b. Socio-technical interaction networks in free/open source software development processes. In *Software Process Modeling*, Acu~na ST, Juristo N (eds). Springer Science, Business Media Inc.: New York, 1–27.

Scacchi W, Jensen C, Noll J, Elliott M. 2005c. Multi-modal modeling, analysis and validation of open source software requirements processes. In *Proceedings of the First International Conference Open Source Software*, Genova, Italy, July 1–8 2005c.

Schach SR, Jin B, Wright DR, Heller GZ, Offutt AJ. 2002. Maintainability of the Linux Kernel. *IEE Proceedings–Software* **149**(1): 18–23.

Sharma S, Sugumaran V, Rajagopalan B. 2002. A framework for creating Hybrid open-source software communities. *Information Systems Journal* **12**(1): 7–25.

Smith M, Kollock P (eds). 1999. *Communities in Cyberspace*. Routledge: London, UK.

Smith N, Capiluppi A, Ramil JF. 2004. Qualitative analysis and simulation of open source software evolution. In *Proceedings of the 5th Software Process Simulation and Modeling Workshop (ProSim'04)*, Edinburgh, Scotland, UK, May 2004.

Stewart KJ, Gosain S. 2001. An exploratory study of ideology and trust in open source development groups. In *Proceedings of the 22nd International Conference Information Systems (ICIS-2001)*, New Orleans, LA.

Truex D, Baskerville R, Klein H. 1999. Growing systems in an Emergent Organization. •*Communications ACM* **42**(8): 117–123.

von Krogh G, Spaeth S, Lakhani K. 2003. Community, joining, and specialization in open source software innovation: a case study. *Research Policy* **32**(7): 1217–1241.

Yamauchi Y, Yokozawa M, Shinohara T, Ishida T. 2000. Collaboration with lean media: How open-source software succeeds. *Proceedings of the Computer Supported Cooperative Work Conference (CSCW'00)*. ACM Press: Philadelphia, PA, December 329–338.

Ye Y, Kishida K. 2003. Towards an understanding of the motivation of open source software developers. *Proceedings of the 25th International Conference On Software Engineering*. IEEE Computer Society: Portland, OR, May 419–429.

Ye Y, Nakakoji K, Yamamoto Y, Kishida K. 2004. The co-evolution of systems and communities in free and open source software development. In *Free/Open Source Software Development*, Koch S (ed.). Idea Group Publishing: Hershey, PA, 59–82.

QUERIES TO BE ANSWERED BY AUTHOR

IMPORTANT NOTE: Please mark your corrections and answers to these queries directly onto the proof at the relevant place. Do NOT mark your corrections on this query sheet.

Queries from the Copyeditor:

AQ1 We have provided the short title ''Understanding F/OSSD Processes. . .'' for the running head for this article. Please clarify whether it is appropriate.

AQ2 Figure 1 are in low resolution (188dpi). Resupply either good quality original or high resolution file if it is not ok for printing

AQ3 This (Carney *et al*. 2003) reference has not been listed in the reference list. Please provide the reference details.

AQ4 Please provide the page range for this reference.

AQ5 Please provide the place of conference for this reference.

AQ6 Please provide the place of publication for this reference.

AQ7 Please provide the place of conference for this reference.

AQ8 Please provide the place of publication for this reference.

AQ9 This (Porter *et al*. 1997) reference has not been cited in text. Please provide the citation for this reference.

AQ10 Please provide the spelt-out form of the journal title.

WILEY AUTHOR DISCOUNT CARD

As a highly valued contributor to Wiley's publications, we would like to show our appreciation to you by offering a **unique 25% discount** off the published price of any of our books*.

To take advantage of this offer, all you need to do is apply for the **Wiley Author Discount Card** by completing the attached form and returning it to us at the following address:

The Database Group
John Wiley & Sons Ltd
The Atrium
Southern Gate
Chichester
West Sussex PO19 8SQ
UK

In the meantime, whenever you order books direct from us, simply quote promotional code **S001W** to take advantage of the 25% discount.

The newest and quickest way to order your books from us is via our new European website at:

http://www.wileyeurope.com

Key benefits to using the site and ordering online include:

- Real-time SECURE on-line ordering
- The most up-to-date search functionality to make browsing the catalogue easier
- Dedicated Author resource centre
- E-mail a friend
- Easy to use navigation
- Regular special offers
- Sign up for subject orientated e-mail alerts

So take advantage of this great offer, return your completed form today to receive your discount card.

Yours sincerely,

V Leaver

Verity Leaver
E-marketing and Database Manager

*TERMS AND CONDITIONS

This offer is exclusive to Wiley Authors, Editors, Contributors and Editorial Board Members in acquiring books (excluding encyclopaedias and major reference works) for their personal use. There must be no resale through any channel. The offer is subject to stock availability and cannot be applied retrospectively. This entitlement cannot be used in conjunction with any other special offer. Wiley reserves the right to amend the terms of the offer at any time.

REGISTRATION FORM
FOR 25% BOOK DISCOUNT CARD

To enjoy your special discount, tell us your areas of interest and you will receive relevant catalogues or leaflets from which to select your books. Please indicate your specific subject areas below.

Accounting [ ]
- Public [ ]
- Corporate [ ]

Chemistry [ ]
- Analytical [ ]
- Industrial/Safety [ ]
- Organic [ ]
- Inorganic [ ]
- Polymer [ ]
- Spectroscopy [ ]

Encyclopedia/Reference [ ]
- Business/Finance [ ]
- Life Sciences [ ]
- Medical Sciences [ ]
- Physical Sciences [ ]
- Technology [ ]

Earth & Environmental Science [ ]

Hospitality [ ]

Genetics [ ]
- Bioinformatics/Computational [ ] Biology [ ]
- Proteomics [ ]
- Genomics [ ]
- Gene Mapping [ ]
- Clinical Genetics [ ]

Medical Science [ ]
- Cardiovascular [ ]
- Diabetes [ ]
- Endocrinology [ ]
- Imaging [ ]
- Obstetrics/Gynaecology [ ]
- Oncology [ ]
- Pharmacology [ ]
- Psychiatry [ ]

Non-Profit [ ]

Architecture [ ]

Business/Management [ ]

Computer Science [ ]
- Database/Data Warehouse [ ]
- Internet Business [ ]
- Networking [ ]
- Programming/Software [ ] Development [ ]
- Object Technology [ ]

Engineering [ ]
- Civil [ ]
- Communications Technology [ ]
- Electronic [ ]
- Environmental [ ]
- Industrial [ ]
- Mechanical [ ]

Finance/Investing [ ]
- Economics [ ]
- Institutional [ ]
- Personal Finance [ ]

Life Science [ ]

Landscape Architecture [ ]

Mathematics/Statistics [ ]

Manufacturing [ ]

Material Science [ ]

Psychology [ ]
- Clinical [ ]
- Forensic [ ]
- Social & Personality [ ]
- Health & Sport [ ]
- Cognitive [ ]
- Organizational [ ]
- Developmental and Special Ed [ ]
- Child Welfare [ ]
- Self-Help [ ]

Physics/Physical Science [ ]

[ ] I confirm that I am a Wiley Author/Editor/Contributor/Editorial Board Member of the following publications:

SIGNATURE: ………

PLEASE COMPLETE THE FOLLOWING DETAILS IN BLOCK CAPITALS:

TITLE AND NAME: (e.g. Mr, Mrs, Dr) ………………………………………………………………………………………

JOB TITLE: ……

DEPARTMENT: ……..

COMPANY/INSTITUTION: ……………………………………………………………………………………………………

ADDRESS: ………

……

……

……

TOWN/CITY: ………

COUNTY/STATE: ………

COUNTRY: ………

POSTCODE/ZIP CODE: ……

DAYTIME TEL: ……

FAX: ……

E-MAIL: ……

YOUR PERSONAL DATA
We, John Wiley & Sons Ltd, will use the information you have provided to fulfil your request. In addition, we would like to:

1. Use your information to keep you informed by post, e-mail or telephone of titles and offers of interest to you and available from us or other Wiley Group companies worldwide, and may supply your details to members of the Wiley Group for this purpose.

   [ ] Please tick the box if you do not wish to receive this information

2. Share your information with other carefully selected companies so that they may contact you by post, fax or e-mail with details of titles and offers that may be of interest to you.

   [ ] Please tick the box if you do not wish to receive this information.

If, at any time, you wish to stop receiving information, please contact the Database Group (databasegroup@wiley.co.uk) at John Wiley & Sons Ltd, The Atrium, Southern Gate, Chichester, West Sussex PO19 8SQ, UK.

E-MAIL ALERTING SERVICE
We offer an information service on our product ranges via e-mail. If you do not wish to receive information and offers from John Wiley companies worldwide via e-mail, please tick the box [ ].

This offer is exclusive to Wiley Authors, Editors, Contributors and Editorial Board Members in acquiring books (excluding encyclopaedias and major reference works) for their personal use. There should be no resale through any channel. The offer is subject to stock availability and may not be applied retrospectively. This entitlement cannot be used in conjunction with any other special offer. Wiley reserves the right to vary the terms of the offer at any time.

Ref: S001W