A Critical Review of Software Engineering Research on Open Source Software Development

Thomas Østerlie
Norwegian University of Science and Technology
thomas.osterlie@idi.ntnu.no

Letizia Jaccheri
Norwegian University of Science and Technology
letizia.jaccheri@idi.ntnu.no

ABSTRACT

This paper asserts that the software engineering (SE) research literature describes open source software development (OSSD) as a homogenous phenomenon. Through a discourse analysis of the SE research literature on OSSD, it is argued that the view of OSSD as a homogenous phenomenon is not grounded in empirical evidence. Rather, it emerges from key assumptions held within the SE research discipline about its identity and how to do SE research. As such, it is argued that the view of OSSD as a homogenous phenomenon may constitute a systematic bias in the SE research literature. Implications of this are drawn for future SE research to avoid reproducing this bias.

Keywords

Software engineering, open source software development, literature review.

INTRODUCTION

Over much of the past decade, researchers have studied the open source software (OSS) phenomenon. After two annual conferences on open source systems (Damiani et al., 2006, Scotto and Succi, 2005), numerous special issues within multiple research fields (Adam et al., 2003, Clarke, 2006, Feller et al., 2002, Scacchi et al., 2006, von Krogh and von Hippel, 2003), as well as several cross-disciplinary paper collections on OSS (Feller et al., 2005, Koch, 2004), it is fair to say that OSS research is maturing as a multi-disciplinary field defined by its object of study, the OSS phenomenon. Researchers have approached the phenomenon from a diversity of angles; among these motivations of OSS developers (Lakhani and Wolf, 2005), social organization of OSS communities (Crowston and Howison, 2005), OSS business models (Karels, 2003), as well as OSS development (OSSD). OSSD is the topic of this paper.

Software engineering (SE) publications have been a major channel for OSSD research. After working with the SE research literature on OSSD for almost a decade, we have grown increasingly concerned with what we find to be a black and white view of OSSD. This paper therefore starts with the following assertion: *the SE research literature describes OSSD as a homogenous phenomenon*. Such a description of OSSD is problematic. Recent empirical studies show great diversity in the phenomenon. Michlmyer et al. (2005), for instance, observe "how greatly development practices and processes employed differ across [OSSD] projects". Yet, describing OSSD as a homogenous phenomenon loses this diversity. While it is reasonable that early research lacks nuances, a more nuanced view is expected as research matures. However, this paper asserts that this is not the case for SE research on OSSD. The following research question is therefore asked: *under what conditions can the view of OSSD as a homogenous phenomenon be made and maintained over time?*

This paper seeks an answer to this question through a critical literature review of published SE research on OSSD. In particular, it seeks an answer to the question by examining how underlying assumptions about both the field of SE as well as about the object of study, OSSD, enables and constrains how SE researchers can describe OSSD. As such, the methodology of this paper is discourse analysis (Phillips and Hardy, 2002). This is therefore not a study of the OSSD phenomenon itself, but rather how it is described in the SE research literature.

This paper makes three contributions. First, it contributes to SE research on OSSD by arguing the case for a potential systematic bias in existing research: that of treating OSSD as a homogenous phenomenon. Second, it motivates the need for diversifying our approach to studying OSSD. Whereas existing reviews of SE research focus on increased scientific rigour and validation of research (Fenton, 1994, Zelkowitz and Wallace, 1998), there has been little focus on how approaches to SE research and the assumptions espoused by these approaches influence the object of study. To the SE research community at large this paper therefore contributes with a possible approach for evaluating the effect research approaches and assumptions have on the object of study. Third, although limited to a survey of SE research on OSSD, the paper may hopefully inspire similar reflections on the implications of research approaches within other parts of SE research.

The remainder of the paper is structured as follows. First the methods and materials that this review is based on are presented. We then ground the assertion that the SE literature treats OSSD as a homogenous phenomenon in

an analysis of the SE research literature. The research question is revisited in the discussion where we show how the view of OSSD as a homogenous phenomenon emerges from three different assumptions. The paper is concluded by drawing implications of the analysis for SE research on OSSD.

METHODS AND MATERIALS

This literature review is approached with discourse analysis. Discourse analysis is a method for studying individual texts for clues to the nature of a discourse. It is the study of how interrelated texts, the practices of their production, dissemination, and reception – collectively labeled the discourse – brings phenomena into being. The phenomenon studied here is OSSD. Discourse analysis examines how language constructs phenomena, rather than how it reflects and reveals them. As such, it embodies a strong constructivist philosophy, and is not just a method but also a methodology.

Although discourses are inscribed and enacted in individual texts, the discourse itself exists beyond these material manifestations: "discourses are shared and social, emanating out of interactions between social groups and complex societaly [sic] structure in which the discourse is embedded" (Phillips and Hardy, 2002). As such, discourse analysis seeks to understand the context within which the discourse is embedded and emerges from. Discourses are therefore analyzed along three dimensions: texts, discourse, and context.

Discourses have no clear boundaries. "We can never study all aspects of a discourse, and inevitably have to select a subset of texts for manageability" (Phillips and Hardy, 2002). The remainder of this section describes our method for selecting this subset of texts to analyze.

Stage 1: Publication Selection

During the first stage, publications outlet for SE research on OSSD had to be identified. Webster and Watson (2002) presents two approaches for identifying relevant literature to review: 1) search through leading journals within the field, and 2) with basis in known literature go backwards by reviewing citations and forwards using research indexes to look for papers citing the known literature. This review follows the first approach, using the selection of six leading journals identified by Glass et al. (2002).

Stage 2: Selection of Texts

Once the journals had been identified, individual publications on OSSD research were identified. The selected journals were accessed through digital libraries. The digital libraries were used to identify individual papers by searching for publications with the keyword 'open source'. The journals are available through different digital libraries. Table 1 lists the journals reviewed with the provider of the digital library. As the digital libraries are continuously updated with new publications, the date of the search is also provided in the table. There are slight variations in the searchable fields supported by the digital libraries. Although these variations have minor impact on the papers identified at this stage, a list of the searchable fields supported by the digital library has been included for reference in Table 1.

Stage 3: Refining the Paper Selection

Searching for the keyword 'open source' in the above digital libraries returned a total of 120 papers. At this stage the subset of papers identified by the digital libraries were manually refined. As some of the digital libraries do not support searching for phrases, some of the returned papers were not on OSSD. Rather, they had been returned

Journal	Date of search	Digital library	Searchable field(s)
Information Software and Technology	February 21 2007	Science Direct (www.sciencedirect.com)	Title, abstract, keywords
Journal of Systems and Software	January 30 2007		
Software Practice and Experience	January 30 2007	Wiley InterScience (www.interscience.wiley.com)	Full text, abstract, article title, author, author affiliation, keywords, references
IEEE Software	January 30 2007	IEEE Xplore (ieeexplore.ieee.org)	Full text, document title, author, abstract
IEEE Transactions on Software Engineering	January 30 2007		
ACM Transactions on Software Engineering and Methodology	January 30 2007	The ACM Digital Library (portal.acm.org)	Title, abstract, author, full text (where available)

Table 1 Journals with corresponding digital libraries

as both the word 'open' and 'source' was found in the searchable fields. To remove such papers from the subset of texts to analyze, the papers were searched for the phrase 'open source'. Papers without this phrase were removed from the subset of texts to analyze.

A number of the papers identified were either a) reports on design research where the product has been released as OSS, b) research where OSS is used as a data set to validate non-OSSD methods or techniques, or c) opinion pieces. As these are not studies of OSSD, they were also removed from the subset of texts to analyze.

52 papers were left after two rounds of refining the subset of texts. This is summarized in Table 2.

Journal	Total papers	Not studies of OSSD	OSSD papers analyzed
Information Software and Technology	7	6	1
Journal of Systems and Software	13	8	5
Software Practice and Experience	15	14	1
IEEE Software	62	23	39
IEEE Transactions on Software Engineering	8	7	1
ACM Transactions on Software Engineering	15	10	5
Total	120	68	52

Table 2 Papers selected

Writing Up the Discourse Analysis

Two interests had to be balanced in writing up this review. With the reader and evaluator in mind, it is important to be as concrete as possible in building a credible case for the assertion that the SE research literature describes OSSD as a homogenous phenomenon. In practice this means making direct references to individual texts. However, it is counter to the goal of the analysis to point out problems, faults, or shortcomings of individual research texts. It is not the goal of the analysis to single out individual researchers and attack their research. Furthermore, discourse analysis is concerned with individual texts only in the way they provide clues to the nature of the discourse.

To balance these two interests, only texts that are often cited by other research and can therefore be considered formative to OSSD research are quoted in the analysis below. The danger of such an approach is that the analysis may seem anecdotal and poorly grounded. Yet, the purpose of discourse analysis is not to bring evidence or establish truths by bringing forth deep or hidden structures in a body of texts. Rather, the analysis in this paper is one of many ways of reading the body of SE research texts on OSSD. As such, the analysis provides a particular lens to view the texts with. The best validation of the analysis is therefore for the reader to approach the same body of literature with the provided lens to determine whether or not the discourse analysis provides a fruitful way of understanding the literature.

ANALYSIS

The purpose of this section is to illustrate in what ways OSSD is described as a homogenous phenomenon in the SE research literature. Four ways are identified: 1) statements about the OSSD model, 2) statements that OSSD is different from SE, 3) studies critically addressing early claims that OSSD produces superior software, and 4) studies of OSS adoption in commercial software development. Each of these approaches is discussed in turn.

Statements About the OSSD Model

Raymond's (1998) seminal paper on describes two different approaches to developing software: the organized cathedral and the buzzing activity of the self-organizing bazaar. The bazaar model of software development has a number of distinguishing characteristics: openness, self-organizing, creative, rapid cycle of releases with frequent incremental updates of the source code (Raymond, 1998). With the advent of the Open Source Initiative (Perens, 1999), the bazaar model of software development is renamed the open source software development model. Espoused in this early period of advocacy literature is the view of OSSD a specific approach to developing software.

Statements about such a specific approach to developing software appears in different forms in the SE literature. Some authors talk of the OSSD model, others about the OSSD cycle, while others talk about the OSS paradigm of software development. While it is sometimes noted that there is variation in this specific approach to developing software, the "basic tenets of OSS development are clear enough, although the details can certainly be difficult to pin down precisely" (Mockus et al., 2002). It is therefore possible to talk about a generic OSSD model (Feller and Fitzgerald, 2002). As such, statements about a specific OSSD model in the SE research literature reproduce the advocacy literature's view of OSSD as a homogenous phenomenon.

A variation of this is to make statements about salient characteristics of OSS or OSSD. SE research paper

frequently describe OSSD as geographically distributed software development, that work is not assigned but undertaken, that there are no plans, that OSS is developed by communities of volunteers, or of there being a particular social organization to OSSD. Statements about salient characteristics with OSSD are made with general significance. They apply to all instances of OSSD, assuming that OSSD is a specific approach to developing software. Such statements about salient characteristics with OSSD espouse the view of OSSD as a homogenous phenomenon.

Making such statements about OSSD as a specific approach to developing software serves two functions in the research literature: to generalize bottom-up and top-down.

By generalizing from the *bottom-up*, single instance of OSSD are made to stand in and represent the larger phenomenon of OSSD. This form of overgeneralization within the OSS research is also observed by Crowston & Howison (2005): "most research on FLOSS [Free/Libre, Open Source Software] has been case studies of particular projects, [and] has so far allowed the perception that there is a distinctive FLOSS organizational pattern and set of practices to go largely unquestioned". To generalize from a single instance of OSSD to the larger phenomenon requires homogeneity of the phenomenon, that all instances of OSSD are comparable.

Top-down generalization is mainly used to motivate research on OSSD. A typical top-down generalization can be formulated as "Our interest in studying this particular instance of OSSD originated in the popularity gained by the open source model in the last few years through the delivery of successful products such as Linux, Apache, and Mozilla". The effect of top-down generalization is to motivate research on a single instance of OSSD by grounding it in the larger phenomenon. By mobilizing well-known successful instances of OSSD, it is assumed that all instances of OSSD are worth studying. Again, this form of generalization assumes homogeneity of the phenomenon; that any instance of OSSD can stand in for the larger phenomenon.

Although bottom-up generalization is most prevalent in early research SE literature on OSSD, the most recent observation is found in a research publication from 2006. Top-down generalizations, however, are in one form or another more prevalent throughout the period of the reviewed literature.

Statements that OSSD is Different From SE

Describing OSSD as different from other forms of software development has been a common theme since the early advocacy literature. To begin with it was the cathedral versus the bazaar (Raymond, 1998), it was hacking as opposed to the mechanical forms of commercial software development (Hannemyr, 1999), and later that OSSD is "different from proprietary, or traditional, or commercial or whatever other forms of software development it is that exist besides [it]" (Crowston and Howison, 2005).

Similar statements about dichotomous relations between OSSD and other forms of software development are reproduced in the SE research literature on OSSD. These statements are made in three ways. The first two ways are direct ways of stating the dichotomous relationship between OSSD and SE. First, as direct statements that OSSD is different from SE. SE is not always referenced directly, but referenced as " the usual industrial style of software development" or "usual methods applied in commercial software development". The implication is clear, however, that OSSD is different from SE.

The second way of placing OSSD in a dichotomous relationship with SE is similar to the above approach, but instead of saying that OSSD is different from SE, authors say that OSSD is not an engineering method. The implied comparison is still OSSD versus SE. All such statements are based in a basic black and white schema: that of OSSD on the one hand and SE on the other.

The third way of making statements that OSSD is different from SE, is indirect. It is indirect in that it makes no reference to SE, "the usual style industrial style of software development", or variations thereof. Instead, the comparison is implied by describing OSSD in terms of work not being assigned, no explicit system-level design, and no project plan, schedule or list of deliverables. OSSD is here characterized by reversing salient characteristics of SE: that in SE work is assigned, there is explicit system-level design, and there is a project plan, schedule or list of deliverables. As such, OSSD is placed in a dichotomous relationship with SE reproducing the two broad categories of OSSD on one hand and SE on the other.

By situating OSSD in a dichotomous relationship with SE implies homogeneity of OSSD; that it is meaningful to situate the phenomenon at large in contrast to SE.

Myth-Busting Studies

Early OSS advocacy literature makes claims about the superiority of OSSD compared to commercial software development. In an effort to develop a deeper and more refined understanding of the OSSD phenomenon, researchers have put these myths about OSSD to the test by comparing OSS with close source software (CSS). These studies aim at providing a more correct understanding of the OSSD phenomenon by challenging empirically unsubstantiated claims. Among these are claims that OSSD compared to CSS produces more maintainable software, simpler designs, software with lower defect density, software with higher quality and reliability, and that OSSD fosters more creativity.

There are two common denominators of these studies. One, the research approach is to generate quantitative measures from products of the software process,

particularly source code and defect reports. Two, these studies compare OSS with CSS either explicitly in the research questions or in discussing the findings.

The earliest of these myth-busting studies date back to 2002, with a predominance of such research published from 2004 and onwards. While most of the tested myths are debunked, the studies' significance in the context of this paper is that they build upon the basic dichotomy of OSS in contrast to CSS. In the process of refining our knowledge of OSSD, these studies reproduce a black and white view of OSSD as a homogenous phenomenon by performing comparisons with the two generic categories of OSS and CSS.

OSS Adoption in Commercial Software Development

A number of studies on OSSD adoption in commercial software development have been published recent years. These studies focus on the adoption of OSS components or OSS tools in commercial software development. Numerous researchers have pointed out the tight relationship of OSSD and commercial software companies (Koru and Tian, 2005). However, the relationship between OSS and commercial actors remains largely unexplored. The studies on OSS adoption therefore aim to broaden our understanding of the OSSD phenomenon by investigating this relationship.

The problem with this literature is two-fold. One, it assumes that OSS is essentially different from commercial off-the-shelf software and therefore requires a unique approach for evaluation. Two, although studying OSS in a commercial setting, these studies do not challenge the view of OSSD as completely different from SE. Instead, they focus on how commercial companies make use of OSSD products. Little, if any, attention is paid to the development of OSS in a commercial context. By omission these studies therefore reproduce the view of OSSD as completely different from commercial software development or SE; a view grounded in the assumption of OSSD as a homogenous phenomenon.

DISCUSSION

The analysis above illustrates the ways in which OSSD is described as a homogenous phenomenon by the SE research literature. The purpose of this section is to address the research question by discussing the conditions under which the statement that OSSD is a homogenous phenomenon can be made and sustained in the context of SE research. As such, this part of the paper broadens the analysis from the discourse itself to its context: SE research.

Assumptions About Software Engineering Research

Glass (2003) observes that "[f]or most of SE's history, authors have eagerly told practitioners what they ought to be doing … [b]ut rarely have those 'ought' been predicated on what practitioners actually are doing". Singer et al. (1997) observe that there is little in the SE research literature about what it is that the software engineers do on a day-to-day basis, the kinds of activities they perform, and the frequency with which these activities take place. While there exist a strain of empirical studies of SE in practice, this has had little or no impact on the mainstream SE research literature. It is therefore unproblematic to state that OSSD is different from SE: OSSD practice does differ from prescriptive models for software development.

SE is a movement of industry and academic actors to professionalize software development by applying engineering to software through the "application of systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software" (IEEE, 1990). The idea of a software crisis is central to this movement. Practically every SE textbook discusses the software crisis, and both SE professionals and researchers keep discussing the continued software crisis (Glass, 2003). Professionalizing software development is the SE movement's answer to the crisis – to a certain extent even its reason to be. The success of OSSD – software developed by volunteers – can be seen as a direct challenge to the very identity of SE, defying the central claim that professionalizing software development will resolve the software crisis.

That the SE research literature maintains the claim that OSSD is different from SE can be interpreted as a way of meeting this challenge. Refuting the general applicability of OSSD outside the specific context where there is a convergence between user and developer can be interpreted as a direct answer to the challenge (Messerschmidt, 2004). Another approach is to characterize OSSD as the inverse of SE as illustrated in the above analysis. Similarly, in comparing OSSD practice with predictive software development models, publishing SE researchers bypassing the problematic issue that the SE research discipline actually knows little about the field they are trying to address: SE in practice.

As such, maintaining the claim that OSSD is different from SE and CSS development serves the purpose of strengthening the SE research discipline. Yet, the effect of this is that OSSD is treated as if it was a single, homogenous phenomenon. And the question remains: how different is OSSD and SE practice?

Assumptions About How To Do Software Engineering Research

The predominance of empirical studies of OSSD reviewed for this paper, are based on either source code measurements or measures extracted from defect tracking and revision control systems. Of the empirical studies reviewed, only three were not based on measurements of products of OSSD. One of these was an ethnographic study (Scacchi, 2004), one a questionnaire survey (Ajila and Wu, 2007), and the third based on undisclosed observational research (Breuer and Valls, 2006). The

dominant approach for SE research on OSSD is therefore to measure the products of the software process.

This approach to studying OSSD grows out of a problem particular to the situation of SE during the 1990s: researchers' observation of a widening gap between software engineering research and practice (Glass, 1994). The software engineering research community was becoming increasingly concerned with its lack of impact on practice. Researchers looked for ways to address this. Tichy et al. (1993) concluded that instead of informing practice, SE research was lacking in quality and thereby becoming less credible for industry. Similarly, in a review of the SE research literature, Fenton (1993) found "very little empirical evidence to support the hypothesis that technological fixes, such as the introduction of specific methods, tools, and techniques, can radically improve the way we develop software systems".

The diagnosis of the problem situation is outlined in a number of surveys of the SE research literature. In a survey of 612 SE research papers, Zelkowitz and Wallace (1998) found that 58.7% of the surveyed papers had no validation of research claims or the validation was based on assertions. Similarly, in a survey of 400 research papers within the broader field of computer science, Tichy et al. (1995) found only 20% of the SE papers devoted more than one fifth or more of the space to research validation. Glass (1994) labels research lacking in validation advocacy research – researchers advocating a new technology without validating its effectiveness over existing technologies or its applicability to practitioners.

A call for increased empirical research and scientific rigour within the software engineering research community rigour rose in response to the problem situation. To bridge the gap between theory and practice, researchers had to move from a research-and-transfer model to an industry-as-laboratory approach (Potts, 1993). Software engineering research needed to better validate its scientific claims (Zelkowitz and Wallace, 1998). The low ratio of validated research had to be rectified for the long-term health of the field (Tichy et al., 1995). However, validation was only one aspect of this increased concern with scientific rigour. Scientific rigour also require better understanding of measurement theory. Fenton (1994) argues that software engineering researchers "must adhere to the science of measurement if it is to gain widespread acceptance and validity".

Quantitative data based on measuring products of the software development process (i.e. source code and data extracted from defect tracking and revision control systems) are well suited for doing comparative research. The myth busting studies make use of this, by comparing OSS and closed source software (CSS) to verify claims made by early OSS advocates that characteristics of OSS differ from CSS. The myth busting studies can be understood as an amalgamation of the OSS and SE discourses in that the scientific approach of empirical SE is applied on open issues raised by the OSS advocacy literature. While the advantage with measurement-based research is the ability to compare, the problem in this case is that the basis of the comparison is the product of OSSD on one side and the product of what is called CSS development on the other. While the studies have been performed with the highest scientific rigour, the amalgamation between the OSS and SE discourses reproduces the very broad distinction of OSS and CSS.

Operating with only two broad categories absolves the researcher from discussing the comparability of the categories. The question is how comparable measures based on products of the software development process are. How comparable is the defect density of a single-user application developed by two OSS developers, the mean number of developers on the SourceForge.org OSS portal, with that of a large multi-team development effort like the Linux kernel, for instance? This is a problem that cannot be met only by "greater discipline and rigour – deeper research, more quantitative data, and more robust cross case analysis" (Feller et al., 2006). The problem itself a product of the research methods employed on OSSD. As such, it beckons a call for increased multiplicity of research approaches.

Assumptions About the Object of Study

Table 3 summarizes the instances of OSSD studied empirically in the analyzed subset of texts. It is striking how a handful of instances of OSSD keep recurring.

OSSD case	Number of studies
Mozilla	4
Linux kernel	4
Other (unspecified)	3
Apache	2
FreeBSD	2
SourceForce.net	1
OpenBSD	1
NetBSD	1
Debian	1
FreshMeat.net	1
KOffice	1
The GNU Compiler Collection	1
OpenOffice	1

Table 3 Summary of OSS cases studied

Have early descriptions of OSSD been turned into prescriptions for choosing instances of OSSD to study? Is that why there are so few cases? Of the cases studied, all comply with the description of OSS projects as mainly volunteer, adhering to the rapid release and fix software development cycle. There are no empirical studies of OSSD in an industrial setting. Studies on OSS adoption are disregarded, as they are not studies of OSSD in an industrial setting, but rather how OSS is used in a commercial setting.

Fitzgerald (2006) raises concerns about the possibility of a broadening gap between the focus of OSSD research and the OSSD phenomenon itself as OSSD is shifting from geographically distributed software development in communities of volunteers towards development by commercial actors. To meet the concern, Fitzgerald (ibid.) proposes that "the open source phenomenon has undergone a significant transformation from its free software origins to a more mainstream, commercially viable form – OSS 2.0".

Is this altogether new? Perens (1999) reports that the Open Source Initiative, and the OSS term itself, originated in a meeting between advocates and the fledgling Linux industry in 1997. The goal of the meeting was to make free software a viable alternative for the mainstream software industry by de-politicizing it. Commercial interests were always strong in the Apache community (Behlendorf, 1999), even prior to IBM deciding to adopt Apache as its official Web server and hiring many of the Apache developers in 1998. Cygnus Solutions is an early commercial actor building upon and driving development of the GNU Compiler Collection (Tieman, 1999). Similarly, RedHat Software, Inc. has developed and maintained OSS for their GNU/Linux distribution since 1995 (Young and Rohm, 1999). In an effort to meet the stiff competition from Microsoft, Netscape released the source code of their web browser as the Mozilla OSS browser in 1998 to differentiate themselves from the competition.

While all empirical studies of Mozilla reviewed for this paper do note the commercial heritage of the source code and that Netscape hires most of the core developers of the Mozilla project, none have studied the relationship between the company and the community. The Mozilla studies are good examples of how the gap between OSSD research and the OSSD phenomenon that Fitzgerald (2006) is concerned about has already developed within SE research on OSSD. Although recent research suggests that commercial interest in OSS is increasing (Ghosh, 2007), this can hardly be argued as a shift in the phenomenon itself. Rather researchers' focus on community-based OSSD has overshadowed the commercial ties, which were never been truly explored. As such, the premise of Fitzgerald's (2006) problem can be understood as a product of existing research's focus on OSSD as geographically distributed, community-based software development.

IMPLICATIONS FOR SE RESEARCH ON OSSD

Through a discourse analysis of the SE research literature, this paper has argued that the assertion that SE research describes OSSD as a homogenous phenomenon is not grounded in empirical research. The research question 'under what conditions can the view of OSSD as a homogenous phenomenon be made and maintained over time?' is answered by situating the OSSD discourse in context of SE research at large. It is argued that the conditions are to be found in assumptions about the SE research field, how to do SE research, and about the phenomenon of OSSD itself. As such, treating OSSD as a homogenous may be a potential bias running throughout the SE research literature on OSSD.

However, this is not a black-and-white picture. Some researchers raise issues about diversity of OSSD practices. However, the full impact of such observations has yet to materialize in SE research on OSSD. This section concludes the paper by drawing implications of this for SE research on OSSD.

Usefulness of the OSS Term

As shown in the analysis, SE researchers often use the term OSSD to make generic statements about a particular approach to software development. However, this is problematic and does to a certain extent assume that OSSD is a homogenous phenomenon. Gacek and Arief (2004) notes that the only common characteristic of OSSD is that software product is released under an license compliant with the Open Source Definition. As such, the usefulness of the term OSSD is limited and espouses a certain view of the phenomenon.

Researchers may avoid this problem by being specific about the instances of OSSD studied instead of relying upon generic descriptions of OSSD. Being specific on the salient characteristics of the studied instances is a basis for discussions on the generalization of research findings. Here are some issues worth focusing on when being more specific about the studied instance of OSSD.

Sizes. How many developers are involved? What kind of software is developed, and how what is its size?

Commercial and/or community. Some OSS projects are completely community driven, other are controlled by companies, and other in turn are community-based with strong commercial ties. Issues worth considering are therefore: Is the case studied community driven or headed by a company? How many of the community members are hired to contribute, and how many are volunteers? What is the distribution of volunteers and hired developers?

Geographical distribution. One of the issues motivating OSSD has been that of studying successful examples of distributed software development. Many cases of OSSD are geographically distributed. Issues worth discussing when writing up research are: What is the geographical

distribution of the developers? Are any groups of developers geographically co-located? How many groups of co-located developers exist? Does the geographical co-location have any impact on the organization of the project? What is the impact of the geographical distribution on coordination within the project? What tools are used for bridging the geographical gap between developers?

Developer demography. While there exist much research on the motivation of OSS developers, we know little about who they are. Apart from Dempsey et al.'s (1999) study of the distribution of contributors to the UNC MetaLab's Linux Archives by studying the domain of their e-mail addresses, there is a distinct lack of research about who OSS developers are. Future research could focus on improving our understanding of who the people developing OSS are.

Implications for Method

We have illustrated how the dominant approach for studying OSSD within SE reproduces the view of OSSD as a homogenous phenomenon. Leading OSSD researchers call for "greater discipline and rigour – deeper research, more quantitative data, and more robust cross case analysis" (Feller et al., 2006). However, the problem is not caused by a lack of methodical discipline or rigour, but rather with the taken-for-grantedness of the phenomenon studied. As such, more cross case analysis may indeed worsen the problem.

Instead, there is a need for diversifying approaches to studying OSSD. The phenomenon needs to be approached with methods that can shed further light on the practice of OSSD, not only on the products of the process. It may be worth looking towards recent studies of OSSD practice within the field of computer supported cooperative work (Ducheneaut, 2005). This research uses ethnographic methods. While studying the product of OSSD may give the impression of homogeneity of the phenomenon, studies of OSSD practice can challenge this by looking at the specifics of practice may reveal if such is really the case.

Implications for Case Selection

There is a poverty of OSSD cases studied, both in the distribution of individual cases but also in that they are all studies of community-based OSSD. There is no research on OSSD in an industry setting. Little attention is paid to the relationship between commercial organizations and OSS communities. How do commercial actors participation in OSS communities impacts on their internal development processes? Future research should address this by studying such instances of OSSD.

Furthermore, an implication of the problem with using top-down generalization for case selection is that the rationale for case selection has to be grounded in salient characteristics of the selected case. Case selection needs to address two questions: What are the salient characteristics of this case that makes it worth researching? What dimensions of the OSSD phenomenon can it shed further light on?

REFERENCES

1. Adam, F., Feller, J. and Fitzgerald, B. (2003) Logicels Libres: Implications pour les Organisations, *Systems d'Information et Management,* 8, 1.
2. Ajila, S. A. and Wu, D. (2007) Empirical Study of the Effects of Open Source Adoption on Software Development Economics, *Journal of Systems and Software,* Forthcoming.
3. Behlendorf, B. (1999) Open Source as a Business Strategy In *Open Sources: Voices from the Open Source Revolution*, (Eds, DiBona, C., Ockman, S. and Stone, M.) O'Reilly & Associates, Sebastapol, CA, 149-170.
4. Breuer, P. T. and Valls, M. G. (2006) Raiding the Noosphere: The Open Development of Networked RAID Support in the Linux Kernel, *Software-Practice and Experience,* 36, 4**,** 365-395.
5. Clarke, D. (2006) Foreword: Special Issue on Free, Libre, and Open Source Software, *Knowledge, Technology & Policy,* 18, 4**,** 3-4.
6. Crowston, K. and Howison, J. (2005) The social structure of free and open source software development, *First Monday,* 10, 2.
7. Damiani, E., Fitzgerald, B., Scacchi, W., Scotto, M. and Succi, G. (2006) In *Second International Conference on Open Source Systems*Springer, Como, Italy, pp. 351.
8. Ducheneaut, N. (2005) Socialization in an Open Source Software Community: A Socio-Technical Analysis, *Computer Supported Cooperative Work (CSCW),* 14, 4**,** 323-368.
9. Feller, J., Finnegan, P., Hayes, J. and Lundell, B. (2006) *Panel: Business models for open source software, Towards an understanding of the concept and its implications to practice*, http://oss2006.dti.unimi.it/slides/businessModelPanel.pdf, Last accessed: January 4 2006.
10. Feller, J. and Fitzgerald, B. (2002) *Understanding Open Source Software Development,* Addison-Wesley, London.
11. Feller, J., Fitzgerald, B., Hissam, S. A. and Lakhani, K. R. (Eds.) (2005) *Perspectives on Free and Open Source Software,* The MIT Press, Cambridge, Mass.
12. Feller, J., Fitzgerald, B. and van der Hoek, A. (2002) Editorial: Open Source Software Engineering, *IEE Proceedings - Software,* 149, 1**,** 1-2.
13. Fenton, N. (1993) How Effective Are Software Engineering Methods?, *Journal of Systems and Software,* 22, 2**,** 141-146.

14. Fenton, N. (1994) Software Measurement: A Necessary Scientific Basis, *IEEE Transactions on Software Engineering,* 20, 3**,** 199-206.
15. Fitzgerald, B. (2006) The Transformation of Open Source Software, *MIS Quarterly,* 30, 3**,** 587-598.
16. Gacek, C. and Arief, B. (2004) The Many Meanings of Open Source, *IEEE Software,* 21, 1**,** 34-40.
17. Ghosh, R. A. (2007) *Study on the: Economic Impact of Open Source Software Software on Innovation and Competiveness of the Information and Communication Technologies (ICT) Sector in the EU (Final Report)*, ENTR/04/112.
18. Glass, R. L. (1994) The Software-Research Crisis, *IEEE Software,* 11, 6**,** 42-47.
19. Glass, R. L. (2003) The State of the Practice of Software Engineering, *IEEE Software,* 20, 6**,** 20-21.
20. Glass, R. L., Vessey, I. and Ramesh, V. (2002) Research in software engineering: an analysis of the literature, *Information and Software Technology,* 44, 8**,** 491-506.
21. Hannemyr, G. (1999) Technology and Pleasure: Considering Hacking Constructive, *First Monday,* 4, 2.
22. IEEE (1990) *IEEE standard glossary of software engineering terminology*, 610.12-1990.
23. Karels, M. J. (2003) Commercializing Open Source Software, *Queue,* 1, 5**,** 46-55.
24. Koch, S. (Ed.) (2004) *Free/Open Source Software Development,* Idea Group.
25. Koru, A. G. and Tian, J. J. (2005) Comparing High-Change Modules and Modules with the Highest Measurement Values in Two Large-Scale Open-Source Products, *IEEE Transactions on Software Engineering,* 31, 8**,** 625-642.
26. Lakhani, K. R. and Wolf, R. G. (2005) Why Hackers Do What They Do: Understanding Motivations and Effort in Free/Open Source Software Projects In *Perspectives on Free and Open Source Software*, (Eds, Feller, J., Fitzgerald, B., Hissam, S. A. and Lakhani, K. R.) The MIT Press, Cambridge, Mass., 3-23.
27. Messerschmidt, D. G. (2004) Back to the user [open source], *IEEE Software,* 21, 1**,** 89-91.
28. Michlmyer, M., Hunt, F. and Probert, D. (2005) In *First International Conference on Open Source Systesm*Genova, Italy, pp. 24-28.
29. Mockus, A., Fielding, R. T. and Herbsleb, J. D. (2002) Two Case Studies of Open Source Software Development: Apache and Mozilla, *ACM Transactions on Software Engineering and Methodology,* 11, 3**,** 309-346.
30. Perens, B. (1999) The Open Source Definition In *Open Sources: Voices from the Open Source Revolution*, (Eds, DiBona, C., Ockman, S. and Stone, M.) O'Reilly & Associates, Sebastapol, CA, 171-180.
31. Phillips, N. and Hardy, C. (2002) *Discourse Analysis: Investigating Processes of Social Construction,* Sage, Thousand Oaks, CA.
32. Potts, C. (1993) Software-Engineering Research Revisited, *IEEE Software,* 10, 5**,** 19-28.
33. Raymond, E. S. (1998) The Cathedral and the Bazaar, *First Monday,* 3, 3.
34. Scacchi, W. (2004) Free and open source software practices in the gaming industry, *IEEE Software,* 21, 1**,** 68-72.
35. Scacchi, W., Feller, J., Fitzgerald, B., Hissam, S. A. and Lakhani, K. R. (2006) Guest Editorial: Understanding Free/Open Source Software Development Processes, *Software Process: Improvement and Practice,* 11, 2**,** 95-105.
36. Scotto, M. and Succi, G. (2005) In *First International Conference on Open Source Systems*Springer, Genova, Italy.
37. Singer, J., Lethbridge, T. C., Vinson, N. and Anquetil, N. (1997) An Examination of Software Engineering Work Practices, *Center for Advanced Studies Conference (CANCON)*,
38. Tichy, W. F., Habermann, N. and Prechelt, L. (1993) Summary of the Dagstuhl workshop on future directions in software engineering: February 17–21, 1992, Schloß Dagstuhl, *ACM SIGSOFT Software Engineering Notes,* 18, 1**,** 35-48.
39. Tichy, W. F., Lukowicz, P., Prechelt, L. and Heinz, E. A. (1995) Experimental Evaluation of Computer Science: A Quantitative Study, *Journal of Systems and Software,* 28, 1**,** 9-18.
40. Tieman, M. (1999) Future of Cygnus Solutions: An Entrepreneur's Account In *Open Sources: Voices from the Open Source Revolution*, (Eds, DiBona, C., Ockman, S. and Stone, M.) O'Reilly & Associates, Sebastapol, CA, 71-90.
41. von Krogh, G. and von Hippel, E. (2003) Open Source Software: Introduction to a Special Issue of Research Policy, *Research Policy,* 32, 7**,** 1149-1157.
42. Webster, J. and Watson, R. T. (2002) Analyzing the Past to Prepare for the Future: Writing a Literature Review, *MIS Quarterly,* 26, 2**,** xii-xxiii.
43. Young, R. and Rohm, W. G. (1999) *Under the Radar: How Red Hat Changed the Software Business - and Took Microsoft by Surprise,* Corriolis Group, Scottsdale, AZ.
44. Zelkowitz, M. V. and Wallace, D. R. (1998) Experimental Models for Validating Technology, *IEEE Computer,* 31, 5**,** 23-31.