Driving Innovation and Business Success in the Digital Economy

Ionica Oncioiu
Titu Maiorescu University, Romania

A volume in the Advances in E-Business Research (AEBR) Book Series

www.igi-global.com

Published in the United States of America by
IGI Global
Business Science Reference (an imprint of IGI Global)
701 E. Chocolate Avenue
Hershey PA, USA 17033
Tel: 717-533-8845
Fax: 717-533-8661
E-mail: cust@igi-global.com
Web site: http://www.igi-global.com

Library of Congress Cataloging-in-Publication Data

Names: Oncioiu, Ionica, 1972- editor.
Title: Driving innovation and business success in the digital economy / Ionica Oncioiu, editor.
Description: Hershey : Business Science Reference, [2016]
Identifiers: LCCN 2016045415| ISBN 9781522517795 (hardcover) | ISBN 9781522517801 (ebook)
Subjects: LCSH: Technological innovations--Management. | Information technology--Management.
Classification: LCC HD45 .D75 2016 | DDC 658.5/14--dc23 LC record available at https://lccn.loc.gov/2016045415

This book is published in the IGI Global book series Advances in E-Business Research (AEBR) (ISSN: 1935-2700; eISSN: 1935-2719)

British Cataloguing in Publication Data
A Cataloguing in Publication record for this book is available from the British Library.

Chapter 3
Taming of "Openness" in Software Innovation Systems

Mehmet Gençer
İstanbul Bilgi University, Turkey

Beyza Oba
İstanbul Bilgi University, Turkey

ABSTRACT

In large-scale Open Source Software (OSS) innovation ecosystems that incorporate firms, a variety of measures are taken to tame the potentially chaotic activities and align the contributions of various participants with the strategic priorities of major stakeholders. Such taming rests on the dual desires of this emergent community of firms to unleash the innovation potential of OSS and to drive it to a certain direction, and it emerges in the form of various organizational activities. By drawing on a sample of large-scale OSS ecosystems, we discuss that methods employed for taming are isomorphic, and overview the emerging strategic pattern for establishing systems of innovation. This pattern involves a related set of practices to balance virtues of OSS community while introducing corporate discipline. In contrast to approaches such as open innovation, which favor isolated reasoning, we present a systemic and historical perspective to explain the continuum in emergence and establishment of strategic patterns.

INTRODUCTION

Collaborative innovation is becoming increasingly commonplace in many industries, ranging from biotechnology to computer hardware/software. Many organizations are struggling with volatility in consumer demands, stiff competition, and try to respond in a creative and agile manner to these market pressures. The consequences, as it appears, frequently involve seeking innovation beyond traditional locus of internal R&D departments. Accordingly, various organizations pursue bilateral or multilateral arrangements to actively cultivate inter-organizational networks of collaborative innovation.

For firms in volatile markets, acknowledging the need to reach out to networks is a first move that open up new challenges. Most of these firms have established practices and strategies based on closed forms of innovation. Some others have experience with 'open innovation', which involves exchanges with

DOI: 10.4018/978-1-5225-1779-5.ch003

other organizations (firms and research institutions), often using established forms of legally binding bilateral arrangements such as licensing. However, network types of organizing defy existing bilateral forms to enable their participants to get organized with one another easily. Firms intending to benefit from networked, collaborative, and open innovation face various challenges regarding how to reason about, interact with, and profit from the networks they are embedded within. Organization of these open innovation practices is a chaotic forefront of innovation, which demands organizational forms of its own, developed through ongoing trials and errors of its current adopters.

Among several other industries, software industry has been one in which attitudes towards collaborative innovation strategies are changing rapidly. It seems that 'adaptation' is replacing 'planning' as the fundamental principle of innovation in software technology development. Within this changing mindset, Open Source Software (OSS), once a marginal movement, is finding its way into mainstream inter-organizational innovation practice in the industry. OSS provides proven methodologies, which stimulate rapid software product evolution at the expense of process predictability (as opposed to the established practices of software engineering), and promote product interoperability which facilitate a more efficient coordination medium (Behlendorf, 1999; Benkler, 2005; Ritala, 2001). Leading players in the computing industry, such as IBM, Apple, and Google, actively cultivate open source innovation networks and/or use software technologies coming out of these networks as components in their key products or operations (West 2003).

However, despite the rhetoric of `openness' surrounding these practices, considerable effort is vested in to tame the `mess'. Within the multitude of often conflicting business agendas in a collaborative innovation system, each participant faces the dilemma of converting collaborative outcomes into competitive advantage. Accessing knowledge resources through networks overcomes rigidities of innovating in isolation (Jorde & Teece 1989; Ring & van de Ven 1994; Leonard-Barton 1992; von Hippel 2006). Yet, an organization expects some predictability, and would like to drive this distributed and emergent, rather than centralized and planned innovation process in a direction that makes sense for its own priorities. While `openness' in OSS is associated with unleashing the bottom-up innovation, blending and aligning it with a business strategy unavoidably faces the dilemma of re-leashing this innovative power so that it is manageable in a 'top-down' way and becomes useful `business-wise'.

In this exploratory study we examine how practices in OSS, which is a non-commercial community organized as "collaborative innovation model" (von Hippel and von Krogh 2003), blends into a hybrid innovation model of the emergent community of commercial firms who adopt these practices. The major theoretical concern of the paper is to explore how the collaboration of two distinct worlds (OSS/bazaar and firms/hierarchies) incrementally changes the established activities and lead to the transformation of a community. In this vein, we investigate six cases (Apache, Linux, Eclipse, Mozilla, GCC, Android) of 'community-led' and 'corporate-led' collaborative innovation projects based on OSS, and attempt to identify common patterns in structures and processes which address the above mentioned dilemma while empowering collaborative, open innovation. In explaining the transformation of the OSS practices, we utilize an activity-based model developed by Jarzabowski (2005). This approach incorporates actors, activities of actors, and the collective structures developed by the shared activities of involved actors. We define the two communities as activity systems comprising distributed actors, shared activities of these actors (recruiting, strategic decision making, licensing, leadership, and quality assurance) and collective structures (governance) developed during the pursuit of these shared activities.

These activities are sites where interactions among actors are promoted and arise from their interactions with the collective structures. Through time these iterated interactions around shared practical activities legitimizes them and provides domains of signification, domination, and legitimization. Furthermore, in this study, practices like governance and appropriation regimes are taken as instruments of mediation between divergent goals and interests of distributed actors. Through the mediation of these practices various constituents reach an agreement about the actions to be taken, and whether these actions are legitimate and acceptable.

Such an approach has been instrumental in understanding the transformation process and especially its duality because it address both the strategic concerns of firms participating in open source software innovation systems, and why certain strategic practices are available for mutual, multilateral acceptance, while others are not. The analysis of data reveals that recent corporate-led open source innovation systems legitimize themselves by drawing from practices developed in community-led systems. Once a set of practices is worked out - by incremental improvements - for the new strategic challenges, they are mimicked within the emerging community. For example, governance mechanisms developed in one innovation ecosystem are reproduced in others without the need for exogenous policies and policy-makers imposing them. In this sense, OSS innovation model today appears to be an avant-garde category of inter-organizational collaborative innovation system which, by the virtue of owing its advantages to aggregation of resources (human and financial) is a valuable case of collaborative, open innovation but does not fit in the related literature which portrays such systems as entities which are shaped by exogenous forces, regulations and policies.

The paper is organized as such; in the following section we present our theoretical framework, which is based on activity-based approach. Then we discuss extant research on the transformation of OSS. The next section covers our research design and data, followed by our findings. Building on six well-known, large-scale OSS-based collaborative innovation projects, we identify common activities and practices that prevail for taming 'openness'. Finally, we discuss how the new practices emerged and transformed the OSS community, with its theoretical and practical implications, followed by conclusions.

THEORETICAL FRAMEWORK

OSS community brings together geographically dispersed actors (i.e. firms and developers) with diverse interests around a common issue: innovation in software. As a situated activity, innovation practice both shapes and is shaped by various activities of the OSS community and its situated actors cannot be decoupled from the context in which they act. In this vein, the study of how innovation practice is constructed within the community must take into consideration the social situation that provides embedded norms of conduct and the micro interpretations of the same social situation by the agents (Jarzabowski, 2005).

This interplay between context (macro) and actors (micro) in the construction of innovation practice reflects the activity, which is the basis of such a reconstruction process and endorses a deeper understanding of the community transformation processes. In such collective action domains like OSS, transformation is a complex social process, which requires collective action of numerous actors. In the process of such a change, besides the purposive actions of multiple actors, structures and processes embedded within the collaboration network play a role in initiating new and disrupting existing practices (Wijen & Ansari, 2007). Thus, we will be utilizing activity-based approach in order to explore how certain innovation

related activities like recruitment of developers, development of new projects, and decision making can trigger and lead the transformation of the existing innovation. Accordingly, we study innovation as a situated activity, which is always constructed and reconstructed by the "practical-evaluative agency" (Emirbayer & Mische, 1998).

This process of transformation can follow various courses: change, recursiveness and stabilization (Jarzabowski, 2005). This paper focuses on the change course of innovation practice and its construction among multiple, distributed actors (mainly developers and firms). Distributed actors while pursuing their own divergent interests reason, question, discuss and challenge the existing innovation practice; as active agents they exploit social contradictions, tensions, conflicts between the system they are part of and the society. This very process of contestation makes the practice itself prone to fragmentation. Thus, various "structural" and "interpretive" actions must be generated to align the interests of distributed actors so that a collective action can be generated.

Such an approach to change deems the study of prevailing community (its structure and philosophy), goal-directed behavior of agents, and the interaction between these two levels. In this framework agents are taken as practical-evaluative, future oriented, and iterative. In the specific case under study, their actions are mediated through governance regimes and licensing arrangements, which enable interaction between them and the community they are part of. These mediating activities also enable coordination of distributed actions and contributions so that collective action is achieved. In this vein OSS is conceptualized as an activity system comprised of actors, their community, and the goal directed innovation practice in which they are involved.

Practices provide continuity in activities by enabling interaction between the actors and collective structures (Jarzabowski, 2005). However, as the constituents of a community (the two communities of community-led and corporate-led projects in our case) interact with another tensions arise due to emergent interpretations of the new activities that are introduced. The same practices, which promote continuity within a software project ecosystem during the interaction with other ecosystems serve as mediators between contradictory views to leverage new patters of activity, change the context and the meaning attributed to the activities.

The focus of this paper is this transformation process in OSS after collaboration among the two communities in software industry. More specifically we study the change in OSS activities and practices after collaboration in various projects with other activity systems that are characterized as firm-centered hierarchies.

Studying Change and Transformation in OSS

Existing research on OSS mostly views the phenomenon from its community roots (see, for example, the summary in von Krogh et al, 2012). Some of the recent studies on OSS focus on the transformation of the various constituents of the ecosystem. By studying processes like development life cycle, product support, licensing and business strategies adopted Fitzgerald (2006) claims that OSS "has metamorphosed into a more mainstream and commercially viable form" (p. 587). O'Mahony and Ferraro (2007) in a multi-method research studied the emergence of a governance system and the introduction of bureaucratic mechanisms in one OSS project ecosystem. Results of their study indicate how through time bureaucratic values were incorporated to the governance of the project community. Further, they claim that by combining elements of both bureaucratic and democratic mechanisms the community was able to develop a governance mechanism where both regimes coexisted. Similarly, West and O'Mahony (2008)

in their comparative study of sponsored and autonomous open source projects found out that governance mechanism utilized by each were different; autonomous projects had a pluralistic governance mechanism and in the sponsored projects major decisions were controlled by the sponsor.

One explanation of this transformation process is the inevitable interaction between open source communities and software companies and the experimentation of new business models. Strategic alliances between corporations and open source communities like Eclipse have been domains for the development of hybrid governance mechanisms (Gençer & Oba, 2011). Corporate involvement in OSS projects has promoted change in the perceptions, development processes, and business models employed by the OSS community (Lundell, Lings, & Lindqvist, 2010). The relationships between firms and OSS communities naturally influence the way of doing business (Dahlender & Magnusson, 2005, 2008; Frey et al., 2011). Another explanation for the transformation is related to work related activities of the developers; employees of corporations during work are allowed to participate in OSS projects with OSS developers (Hauge, Sorensen, & Conradi, 2008). This daily mundane activity of work related exchange creates a milieu where different worlds experience each other and carry their experiences to their own realms. Bergquist et al. (2011) claim that free and open source software (FOSS) has transformed from an ideological movement to a commercially viable form of software development and this process of transformation has been characterized by various justifications of utilizing open source at different time periods. Rajala, Westerlund and Möller (2012), discuss market oriented OSS business model transformation of MySQL open innovation activity; firms operating in the field of OSS require adaptation to community values and respond to the needs of customers so that community resources are mobilized to enhance innovation capabilities.

This paper, analyzes the transformation of OSS community practices that resulted from the interaction with other software ecosystems within the industry that have a divergent rationale for recruitment, quality assurance, strategic decision making, intellectual property, and governance regimes. In terms of exploration, OSS communities are not controlled hierarchically by the companies with which they establish various forms of cooperation, and there are no binding contracts between the companies and community members. Similarly, exploitation of economic value from the jointly created products by various legal mechanisms is against the basic assumptions of the OSS community. Thus, as we argue, the transformation is not induced by business models or legal adjustments but rather is an inherent outcome of the interaction of two different ecosystems.

RESEARCH METHODS

In order to qualitatively explore the transformation of open source software communities and their practices we have selected a sample of six software projects that are still in operation. Selection of cases was guided by theoretical sampling process on the basis of difference in business models, size, production capacity, and founding date. In so doing, we have included "transparently observable" (Pettigrew, 1990) cases from two major categories of business models: community-led and corporate-led OSS projects. Some of these projects started as small non-commercial projects led by a small group of developers, but later meet corporate interest while some others are initiated by one or a group of firms, and later enjoyed support of individuals from OSS community. These six cases, Apache, Linux, Eclipse, Mozilla, GNU Compiler Collection (GCC), and Android were regarded as appropriate to study the transformation process.

Data for the cases were collected from documents archives. We have analyzed documents published by projects on their websites including statements, announcements, software release logs, developer guidelines, and committer policies. For each case we collected longitudinal data about each issue included in our model. To complement and provide an outsider view, besides publicly available company statements, we have collected news in journals, blogs, and scholarly articles. Collecting data from different sources and opposing opinions counteracted possible bias that can result from relying on a single and coherent source.

The qualitative data set we developed is utilized for a thematic analysis, which started with broad categories and then reduced to key issues under investigation. In order to explore how governance regime has transformed we focused on decision-making process. Our central question was designed to analyze how and by whom decisions are made in software project communities and how this decision making process differed in time. The first part of our main question, how decisions are made reflected the collective structures (units, foundation, lieutenants and committers) and the styles (top-down or grass-roots) developed over time in relation to governance. Major decisions in OSS projects are related to:

1. Setting design goals of software products by selecting among proposed features,
2. Creating sub-projects as needed (i.e. re-bundling the existing features), and
3. Promoting some contributors to 'committer' status who has the right to change or accept changes to the master copy of a software source code.

These decision categories are related to human resources (staffing), quality assurance, and structuring of activities. Thus, as goal-directed practical activities we have studied staffing, recruitment, structuring, and quality assurance because these activities are related to:

1. Coordination of responsibilities/outcomes,
2. Making common sense and appealing to the participants for collective decision making, and
3. Influencing the goals or outcomes of participants (firms and developers) that promote the reconstruction of existing arrangements.

The data analysis of this cross-case study is comprised of three phases. Firstly, we analyzed each case to identify the related activities and came up with a chronological list of such elements, which involved type of decision, actors involved and the type of structure. With this within-case analysis we were able to develop an in-depth understanding of the transformation process that is revealed in each case and expand the extant arguments. Later this list was redesigned to include only those that are common in all the cases comprising the sample. In the second phase, we examined each element across all the cases, to identify how and why it has emerged, who were involved and what were the subsequent outcomes. During this phase we also checked how activities and actors interact with each other and how these patterns differ between cases. And finally, at the third phase we analyzed how these activities and actors contributed to the transformation of OSS, especially in the governance regime.

FINDINGS

In this section, we present the results of our cross-case analysis of the data. The documentation of decision paths revealed six generic outcomes that are commonly observed in all cases: licensing, governance structure, acceptance of new (sub) projects, human resources (promotion), organizational leadership, and quality assurance. The comparison of community-led and corporate-led projects in these activities is given in Table 1.

Table 1. Identification of activities

Project	Overview
Apache (community-led)	• Web server software which has a *liberal license* allowing use in commercial products. • A community project since its start in 1999 but enjoyed a lot of corporate support early on from firms like IBM who use the technology in some of their products. • Governed by the Apache Group which later became a foundation. • New activities are put through an *incubation* process for taming "providing guidance and support to help each new product engender their own collaborative community, educating new developers in the philosophy and guidelines for collaborative development as defined by the members of the Foundation"
Linux (community-led)	• An operating system whose development is led by a vast global community, with a public license (General Public License, GPL) limiting commercialization. But it is used as infrastructure element by many firms. • Some key developers were later hired by leading companies in the industry. It lacks any formal governance body, and led by its originating leader Linus Torvalds since 1991, and has a closed and small leadership team. • Accelerating corporate contributions have prompted creation of a foundation in 2007 dedicated to fostering the technology, and hosts several events where influential community members are brought together.
Eclipse (corporate-led)	• A software development platform initiated by IBM in 2001. • The alliance has later become a foundation which oversees the development ecosystem in a more transparent way. Foundation members are leading firms of the computing industry who pay dues and dedicate developers to the project. • Contributing individuals are promoted to the committer status by existing committers, provided that they demonstrate 'discipline and good judgment'. • New projects are put through an incubation process, similar to Apache case. • The liberal license of Eclipse software was later refined by the foundation and made compatible with GPL to facilitate reuse in appropriation, along with changes in code acceptance policies.
Mozilla (corporate-led)	• A project with web browser and e-mail software. • Initiated by the firm Netscape upon its failure to compete with Microsoft's Internet Explorer. • Its license has gone through several changes from one that explicitly favored Netscape, to a more standard liberal license compatible with the broader open source ecosystem (de Laat, 2005). • Mozilla considers itself as a "hybrid organization, combining non-profit and market strategies to ensure the Internet remains a shared public resource". • Uses an incubation process similar to Apache for taming new projects.
GCC (community-led)	• The GNU Compiler Collection is a piece of software which is used for compiling much other software for a variety of platforms. • Is licensed with GPL. • Its development, started in 1987, was confined within the GNU team led by Richard Stallman, a prominent figure of the open source movement. • As it became a fundamental technology for many firms targeting the UNIX platform, a steering committee was formed in 1999 with representatives from leading firms and universities, but the project leadership emphasizes that the committee members represent communities, not their employers.
Android (corporate-led)	• An open software platform for mobile devices had an impact in the market in only three years following its announcement in November 2007. • The software is licensed using the liberal OSS license used in the Apache project. • Although promotion policies are transparent, these are less meritocratic compared to more mature projects like Eclipse; for example it is only the project leaders (who happen to be Google employees) who promote others, etc. • It is unclear how new projects are approved.

Similar Practices Across the Cases

One common element among the community and corporate-led projects is the human resource activity based on meritocratic promotion of developers. Meritocratic promotion is seen as an important basis of OSS model and seems to be fully adopted in corporate-led OSS projects as well. Besides somewhat exceptional case of Android, both community-led and corporate-led projects adopt voting-based schemes for developer promotion. Contributors who demonstrate skills and adapt to community norms are voted into ranks by existing committers. Norms are (re)articulated by existing meritocracy implementation and thus become an important element of taming within corporate-led collaborative projects.

Quality assurance methods such as software testing and peer review are very important to reduce the adverse effects of defects in software engineering. In terms of quality assurance, OSS is commonly praised as hosting processes in which software quickly evolves into high quality products despite the lack of central control which empowers it. The general quality assurance strategy in OSS has been to transfer testing and bug reporting activities to software user base. Many OSS projects make frequent releases of 'testing' versions of software, which in turn put into use by a large group of enthusiasts and those who are in need of new features not yet offered in 'stable' releases of software. Existing evidence (Zhao & Elbaum, 2003) indicate that frequent releases is often the case for OSS projects for promoting high quality products. In large OSS projects such as Gnome, Mozilla, and Debian, quality assurance has become a part of the community awareness. These projects explicitly stated

1. Their quality goals such as to improve the product where it is needed, and to keep the quality of the distribution as high as it should be,
2. Who the quality participants are, and
3. What are typical quality assurance actions and to which part of software they should be applied.

The development guidelines of the projects studied indicate that the OSS style of quality assurance is applied in all types of projects. However there are differences related to their usage and organization. All the cases followed OSS pattern of releasing frequently. Community projects such as Linux and GCC, and to some degree corporate-led projects like Eclipse and Mozilla, follow the pattern of releases marked as 'testing' or 'release candidate'. These releases are targeted to the user community who are willing to take the risk of using buggy software (in many cases for the sake of accessing new functionality) and provide feedback/bug-reports which in turn assist the developers to carry the software quality to a more mature level. Additionally, both types of projects have clear -even formal- test plans and procedures. In the case of Apache, which is a collection of numerous sub-projects, the quality assurance activities reveal some variation which depends on the nature of the developer community (e.g. size of the development team, type of project sponsorship, project complexity, and release policies). At the other extreme, in Android, the development of the core system is pretty much closed within Google and the quality assurance activities appear to be similar to the ones in closed innovation systems, i.e. they are shaped by the fact that the product goes live on millions of mobile devices of non-tester users. Although they make frequent releases much like OSS style, they work hard (behind closed doors) to ensure that software developed is stable. On the other hand the Android ecosystem consists of thousands of other, higher level pieces of software (Android applications), which are developed by strategic partners and are not subject to such centrally administered quality assurance policies and procedures.

One noteworthy fact in quality assurance activities is that as the some projects become more commercially oriented, they incorporate additional steps to quality assurance process to ensure that contributions meet the license framework and legal requirements related to the commercialization of the software developed. This is the case with Apache and Eclipse. These projects ensure the code contributions go through several steps related to software quality assurance (automated tests, dependency checks, etc.) and also license sanitation. Overall the case of quality assurance in both corporate-led and community-led projects seem to retain proven OSS methods (much like the case for promoting developers) and extend it with context specific requirements.

Differences Between the Cases

Strategic decisions related to the creation of new sub-projects is handled in a more 'transparent' and 'planned' way in corporate-led projects where the process is given names as 'incubation period/process'. In Apache and Eclipse examples, this period involves tempering of the new project and its leadership so that the software development process meets certain quality standards. In community-led projects, on the other hand, all is decided by meritocratic leadership. There may even be cases where leadership makes decisions despite strong opposition (Gençer et al., 2006) which is hard to imagine in case of corporate-led projects.

The difference in property regimes and corresponding licensing schemes reflect the difference of concerns with wider adoption (in community-led type) and suitability for use in commercial products (in corporate-led type). The experience seems to have shown that replacement of intellectual property hostile regime of free software movement with more liberal licenses is generally welcomed. Despite minor variations liberal licenses (with archetypes such as Apache and Mozilla licenses) seem to be the established norm for new corporate-led OSS projects, whereas more restrictive and viral licenses such as GPL is the norm for community-led projects.

In comparing the two types of projects, it is interesting to note that community-led projects are less transparent in terms of governance regime employed. This situation may look somewhat counter-intuitive at first, but there are strategic advantages of transparent governance in an inter-organizational setting. Such transparency serves well for directing community attention and goal setting. Fitzgerald (2006), notes how such transparency has been utilitarian for avoiding the strategic planning vacuum, in which software development followed any 'itch worth scratching'. Corporate-led projects aim to be inclusive and flexible in their planning process, but nevertheless such planning vacuum is avoided by careful outlining of targeted software features and development road-maps, and making them widely available. Hence, using transparency as an instrument has been valuable for directing community attention and setting goals effectively.

Similar to governance mechanisms, organizational leadership is more transparent and works through formal bureaucracy in corporate-led projects community, as in the example of Eclipse where each major participant dedicate certain amount of resources (human and financial) to the project and have seats in decision-making bodies. On the other hand, corporate influence in community-led projects, when it exists, is evidenced by having seats in governance committees. Generally, as Watson et al. (2008) notes it's rather the amount of contribution rather than formal ownership that is the basis of control and influence in community-led projects.

DISCUSSION

A summary of similarities and differences of cross-case analysis is given in Table 2. The findings of our study provide a picture of what is being 'tamed' in business adoption of OSS, and in what direction the process heads. Most noticeable OSS element that is left intact in this taming process is the meritocratic promotion of developers. Watson et al. (2008) note how OSS enables companies to access high quality talent base while reducing risks associated with recruitment. Appreciation of a large group of software developers is a reliable criteria for screening new talent. Thus, retaining the meritocratic promotion does not only ensure software quality, but also provide access to relevant talent pool. In similar vein, adjustments to OSS licensing (see de Laat 2005 for a review) are designed in such a way that while it attracts high quality and increasing volume of contributions, at the same time it allows these contributions to be used in commercial products. Use of more liberal licenses in the corporate-led OSS community is a key instrument in capturing collaboratively created value out of new software.

On the other hand, formal governance practices and goal setting activities are specific to corporate-led realm of OSS and it appears to be still experimental. Some popular success stories like Eclipse are likely to set emerging norms in this respect. Furthermore, comparison of the Android case with Eclipse and Apache suggest that developing a supportive base for collaborative innovation deems increasing transparency in governance structures and practices. Such transparency not only improves trust among partners, but also makes the process more predictable. Additionally, it allows participants to cast a direction in terms of goals and outcomes of the collective process, thus contributing to the taming of OSS. In a business environment where there are established bilateral forms such as joint ventures, such multilateral forms of governance are rather new.

Theoretical Implications

This phenomenon further highlights the need for ontological commitment to network in inter-organizational collaboration research, since the networks start to become concrete entities with some sort of controlling bodies and norms of their own. Despite the awareness about advantages of innovation networking, the business environment is at an infancy phase in terms of institutionalization of collaborative innovation

Table 2. A comparison of corporate-led and community-led OSS projects

	Community Led Projects	Corporate Led Projects
Licensing schemes	GPL, targets largest adoption	Liberal, balances adoption and appropriation
Governances regimes	Not formal, meritocratic	Foundation/bureaucratic, transparent
Strategic decision making process	Not transparent, left to leadership	Well defined incubation processes
Developer promotion	Meritocratic promotion of developers	Meritocratic promotion of developers
Organizational leadership	Imposed by hiring lead developers, or through committees	Imposed through formal bureaucracy
Quality assurance	Community powered	Community powered, with additional measures for legal issues

networks (Whittington et al., 2011). Collaborative innovation is prone to a variety of problems; the initiation of inter-firm relations relies on a variety of antecedent factors (Oba & Semerciöz, 2005), and development of relations and mutual trust that sustain them is hard to predict or control (Ring & van de Ven, 1994). Developing collaborative relations entail sharing critical knowledge (Gächter et al., 2010) and taking risks as committing to relation specific investments.

Our study focused on the emergence and role of practices and structures that are associated with taming of collaborative innovation networks based on OSS model. We have used six cases of community-led and corporate-led projects, and attempted to identify common elements in both types of projects as well as differences, after collaborating with each other. We contend that community-led and corporate-led OSS communities differ: (1) practices and formal arrangements for taming are more explicit and transparent in corporate-led community, (2) meritocratic basis of developer promotion is retained in both, and, (3) the means of asserting influence by corporations is different in both, but considered legitimate as far as the meritocratic basis is retained.

Studies on collaborative innovation mainly focus on the ability of a firm to attract ideas and resources from external actors and appropriate benefits of innovation. As indicated by Langlois (1990) firms are pursuing ways to improve innovation performance beyond the hierarchical structures and thus, mix of new organizational forms are emerging. Studies such as van der Linden et al. (2009) present economic frameworks within which the OSS innovation model is being employed by companies. During the last decade, research reports such as van der Linden et al. (2009) and Fitzgerald (2006) appealed to mainstream practice, unlike the preceding decade where OSS-related literature was rather marginal. Such literature offers a retrospective account of the open source experience in economic terms such as value chain and differentiation. While we lack any extensive evidence at the global level, there are regional studies whose results support that adoption of OSS model is on the rise (Hauge et al., 2008; Nikula & Jantunen, 2005).

Accordingly, OSS is an appropriate case to study how due to certain endogenous and exogenous pressures, innovation regimes can incorporate some elements of other innovation regimes. In the background of these developments is the increasing speed of innovation, driven by variety and volume of demand, high connectivity of devices, and high degree of transferability of information. This makes it hard for even the biggest players to control or supply complex products on their own. When a complex system such as the software industry is on the verge of a fundamental transformation, various marginal practices can become common as they fit well into the new environment. Furthermore, these market pressures trigger creation of software ecosystems where different types of innovation and governance regimes had to cooperate in order to cope with the emergent changes. Innovation regimes build on in-house software production and revenue generation by patented products gradually has adopted elements of the OSS community. Traditionally, the major concern of firms operating in this innovation regime was whether to buy or make the software, and recently this problem has been transformed to collaborate or not; and if the decision is to collaborate then how the relations with the partners should be governed so that efficiency and appropriation would be achieved. The consequences of such a transformation unavoidably lead to transformation of the OSS innovation regime as well. Some authors have already pointed to fundamental shifts in OSS processes, and noted that these indicate alteration of ground rules in both the OSS movement and the software industry (Fitzgerald, 2006). Fitzgerald also explains how 'OSS 2.0' balances "a commercial profit value-for-money proposition while still adhering to acceptable open source community values" (2006 p: 588).

Our findings extend this argument and present the organizational elements that are associated with the shifts in the open source process towards more corporate involvement. There appears to be an emergent, stable pattern of proven mechanisms, strategies, and processes. Thus, the transformation indicated by Fitzgerald's OSS2.0 is institutionalized into common workable governance mechanisms that emerged from the daily activities, and enabled participant market actors to set-up collaborative software ecosystems. Furthermore, emergence of such innovation and governance practices seems to be isomorphic; since they make common sense in most situations and they are practical, actors do not seek other ways of doing things, and avoid introducing disturbance of their respective systems.

Managerial Implications

We call for due attention to the state of the change and transformation in the software industry. We suggest that emergence of the transformation mentioned by Fitzgerald (2006) has already been institutionalized and legitimated, hence will soon lead to faster adoption of OSS strategies by a wider community. The new hybrid innovation regime incorporates activities and mediating practices of previously existing innovation regimes. Such a new design has developed from existing patterns and practices, in incremental steps such that each modification is recognizable and makes sense to stakeholders. Yet, the emergent attractor is radically different from its dual origins (of OSS and proprietary models). Its elements are different from those common in corporate hierarchies; rather than targeting a top-down control or planning in the strict sense, they are instruments of taming. It is through these elements that an corporate-led OSS continues to be an emergent system which retain advantages of bottom-up innovation, while at the same time its major stake-holders can "invite it to emerge" (Jelinek, 2004) in certain ways rather than others, hence avoiding the planning vacuum.

Greater acknowledgment of the need for collaborative innovation is one side of the transformation in the software industry, and OSS seems to be the organizational model of choice in many sub-contexts of the industry. Its emergence triggered by the success of popular community products such as Linux and Apache, but its true admittance to business strategies went through a series of experiments, which in turn transformed OSS practices as well. As far as this emergent combination of solutions work, they can be the first choice of future collaborative software innovation ecosystems in software industry. Nokia's Symbian project might as well continue to be as innovative as Android, but the players of the industry seem to be connected much more easily to OSS model at the moment. Phrasing the current situation rather dramatically, OSS makes a lot of sense, and makes all others look like nonsense.

Software industry innovation practices indicate that free software and open source software are settling in which focus shifts to co-endowment rather than stressing their differences. The community now enjoys more interest in terms of resources provided by corporate ventures, while the corporate circles keep finding ways to turn community experience to viable business practices. Benefits of firms are not much different (e.g. sharing costs of innovation by the help of aggregation), but more than that they have a fundamental interest in capturing value that is created during their collaboration with OSS ecosystems; ensuring legal conditions (software licensing) for capturing value they have helped to create, or assuring quality of software partially built by others, but they will be liable for once it is sold. Therefore a fundamental shift in the requirements and expectations of the practitioners calls for a change in OSS innovation model as well.

CONCLUSION

This study focused on the innovation related practices of two different OSS communities: corporate-led and community-led. We have used a sample of six cases to identify similarities and differences in practices. In so doing we were able to identify various structural properties and practices that emerged in both communities while they collaborated in innovation. Thus, we argued that while these two different worlds collaborated they adopted and transformed some of the practices and structures from each other and retained some others. Our findings indicate how innovation based cooperation changed the dominant logic prevailing both communities and lead to convergence.

REFERENCES

Behlendorf, B. (1999). Open Source as a Business Strategy. In C. DiBona, S. Ockman, & M. Stone (Eds.), *Open Sources: Voices from the Open Source Revolution* (pp. 149–170). O'Reilly.

Benkler, Y. (2005). Coase's penguin, or, linux and the nature of the firm. In R. A. Ghosh (Ed.), *CODE: Collaborative Ownership and the Digital Economy* (pp. 169–206). Cambridge, MA: The MIT Press.

Bergquist, M., & Ljungberg, J. Bertil Rolandsson. "A Historical Account of the Value of Free and Open Source Software: From Software Commune to Commercial Commons. In S. Hissam, B. Russo, M. de Mendonça Neto, & F. Kon (Eds.), *Open Source Systems: Grounding Research* (pp. 196–207). Boston: Springer.

Dahlander, L., & Magnusson, M. (2008). How do firms make use of open source communities? *Long Range Planning*, *41*(6), 629–649. doi:10.1016/j.lrp.2008.09.003

Dahlander, L., & Magnusson, M. G. (2005). Relationships between open source software companies and communities: Observations from Nordic firms. *Research Policy*, *34*(4), 481–493. doi:10.1016/j.respol.2005.02.003

de Laat, P. B. (2005). Copyright or copyleft?: An analysis of property regimes for software development. *Research Policy*, *34*(10), 1511–1532. doi:10.1016/j.respol.2005.07.003

Emirbayer, M., & Mische, A. (1998). What is agency? *American Journal of Sociology*, *103*(4), 962–1023. doi:10.1086/231294

Fitzgerald, B. (2006). The transformation of open source software. *Management Information Systems Quarterly*, *30*(3), 587–598.

Frey, K., Lüthje, C., & Haag, S. (2011). Whom should firms attract to open innovation platforms? The role of knowledge diversity and motivation. *Long Range Planning*, *44*(5), 397–420. doi:10.1016/j.lrp.2011.09.006

Gächter, S., von Krogh, G., & Haefliger, S. (2010). Initiating private-collective innovation: The fragility of knowledge sharing. *Research Policy*, *39*(7), 893–906. doi:10.1016/j.respol.2010.04.010

Gençer, M., & Oba, B. (2011). Organising the digital commons: A case study on engagement strategies in open source. *Technology Analysis and Strategic Management*, *23*(9), 969–982. doi:10.1080/09537325.2011.616698

Gençer, M., Oba, B., Özel, B., & Tunalıoğlu, V. S. (2006). Forking: The GPL coherent technology for flexible organizing in foss development. European Group of Organizational Studies 2006 Colloqium, Bergen, Norway.

Hauge, O., Sorensen, C. F., & Conradi, R. (2008). Adoption of open source in the software industry. In B. Russo, E. Damiani, S. Hissam, B. Lundell, & G. Succi (Eds.), Open Source Development, Communities and Quality. Springer.

Jarzabkowski, P. (2005). Strategy as practice: An activity-based approach. *Sage*.

Jelinek, M. (2004). Managing Design, Designing Management. In R. Boland & F. Callopy (Eds.), *Managing as Designing*. Stanford.

Jorde, T. M., & Teece, D. J. (1989). Competition and cooperation: Striking the right balance. *California Management Review*, *31*(3), 25–37. doi:10.2307/41166568

Langlois, R. N. (1990). Creating external capabilities: Innovation and vertical disintegration in the microcomputer industry. *Business and Economic History*, *19*, 93–102.

Leonard-Barton, D. (1992). Core capabilities and core rigidities: A paradox in managing new product development. *Strategic Management Journal*, *13*(S1), 111–125. doi:10.1002/smj.4250131009

Lundell, B., Lings, B., & Lindqvist, E. (2010). Open source in Swedish companies: Where are we? *Information Systems Journal*, *20*(6), 519–535. doi:10.1111/j.1365-2575.2010.00348.x

Nikula, U. (2005, July). Quantifying the Interest in Open Source Systems: Case South-East Finland Sami Jantunen. *Source*, 192-195.

O'Mahony, S., & Ferraro, F. (2007). The emergence of governance in an open source community. *Academy of Management Journal*, *50*(5), 1079–1106. doi:10.5465/AMJ.2007.27169153

Oba, B., & Semercioz, F. (2005). Antecedents of trust in industrial districts: An empirical analysis of inter-firm relations in a turkish industrial district. *Entrepreneurship and Regional Development*, *17*(3), 163–182. doi:10.1080/08985620500102964

Pettigrew, A. (1990). Longitudinal field research on change: Theory and practice. *Organization Science*, *1*(3), 267–292. doi:10.1287/orsc.1.3.267

Rajala, R., Westerlund, M., & Möller, K. (2012). Strategic flexibility in open innovation–designing business models for open source software. *European Journal of Marketing*, *46*(10), 1368–1388. doi:10.1108/03090561211248071

Ring, P. S., & van de Ven, A. H. (1994). Developmental processes of cooperative interorganizational relationships. *Academy of Management Review*, *19*(1), 90–118.

Ritala, P. (2001). Coopetition Strategy - When is it Successful? Empirical Evidence on Innovation and market performance. *British Journal of Management*.

van der Linden, F., Lundell, B., & Marttiin, P. (2009). Commodification of industrial software: A case for open source. *Software, IEEE.*, *26*(4), 77–83. doi:10.1109/MS.2009.88

von Hippel, E. (2006). *Democratizing Innovation*. MIT Press.

von Hippel, E., & von Krogh, G. (2003). Open source software and the "private-collective" innovation model: Issues for organization science. *Organization Science*, *14*(2), 209–223. doi:10.1287/orsc.14.2.209.14992

von Krogh, G., Rossi-Lamastra, C., & Haefliger, S. (2012). Phenomenon-based research in management and organisation science: When is it rigorous and does it matter? *Long Range Planning*, *45*(4), 277–298. doi:10.1016/j.lrp.2012.05.001

Watson, R. T., Boudreau, M. C., York, P. T., Greiner, M. E., & Wynn, D. Jr. (2008). The business of open source. *Communications of the ACM*, *51*(4), 41–46. doi:10.1145/1330311.1330321

West, J. (2003). How open is open enough?melding proprietary and open source platform strategies. *Research Policy*, *32*(7), 1259–1285. doi:10.1016/S0048-7333(03)00052-0

West, J., & O'Mahony, S. (2008). The role of participation architecture in growing sponsored open source communities. *Industry and Innovation*, *15*(2), 145–168. doi:10.1080/13662710801970142

Whittington, R., Cailluet, L., & Yakis-Douglas, B. (2011). Opening Strategy: Evolution of a Precarious Profession. *British Journal of Management*, *22*(3), 531–544. doi:10.1111/j.1467-8551.2011.00762.x

Wijen, F., & Ansari, S. (2007). Overcoming inaction through collective institutional entrepreneurship: Insights from regime theory. *Organization Studies*, *28*(7), 1079–1100. doi:10.1177/0170840607078115

Zhao, L., & Elbaum, S. (2003). Quality assurance under the open source development model. *Journal of Systems and Software*, *66*(1), 65–75. doi:10.1016/S0164-1212(02)00064-X