Journal of Leadership and Management®
An International Journal of Strategic, Behavioral and Public Aspects of Leadership in Management
Cabell's Listed
A journal of the Institute of Leadership and Management Association, PO Box 564, Douglassville PA 19518. 484-332-3331. https://www.jleadershipmanagement.org

RESILIENT OPEN INNOVATION THROUGH INFORMATION TECHNOLOGIES

U. Yeliz Eseryel,
East Carolina University, Greenville, North Carolina, USA

Abstract

Companies increasingly move towards open innovation strategies. Many organizations adopt disruptive business models such as by emulating open source software development processes or by adopting open source software products. Disruptive business models and management innovation are increasingly built around information technologies. Open innovation, by definition, requires the incorporation of many internal and external knowledge sources into the organizational innovation process. Thus, information technologies are natural enablers and crucial facilitators of open innovation. Yet, the organizational innovation literature of the last 30 years almost completely ignores the role of information technologies in innovation process, with the exception of the use of ideation tools. Recently, the literature in the field of information systems started to fill in this research gap. Yet, little is known about the role of information technology in enabling innovation in different contexts such as (open source) software development. This study attempts to address this knowledge gap by investigating the open innovation process in the open source software development communities. A key challenge of open source software development community success is the existence member and leader turnover. Such turnover may be detrimental to a community's continuity and innovation. Yet, turnover is the norm rather than the exception in open source communities, since these community members are voluntary members. This study investigates how open innovation communities such as open source software development communities innovate through information technologies despite changes in their membership and leadership. This investigation is conducted through comparative case study of two very innovative open source software development communities, which differ in their membership change versus stability.

Keywords: *Open Innovation, Open Source Software Development Communities, Knowledge Management, Information Technology Artifact*

Early Online Publication, Nov 1, 2023. Volume: XX, Issue: XX

INTRODUCTION

The speed with which technology evolves and the increased globalization changed the nature of competition in industries, and how companies organized their research and development processes (Trott & Hartmann, 2009). One example of how companies have changed their approach to work is a move towards open innovation strategies to enable shorter innovation cycles (Gassmann & Enkel, 2001). Open innovation strategies enable organizations to incorporate innovation from various external knowledge sources, including their stakeholders, such as the users of their products. Open innovation is the process of systematically encouraging and exploiting a wide range of internal and external knowledge sources for accelerating innovation (Chesbrough, 2003; Conboy & Morgan, 2011). Often, information technology (IT) is a crucial facilitator of open innovation (Chesbrough, 2003; Nambisan & Sawhney, 2007). Openness, by definition, requires companies to increase the number of external knowledge sources into their innovation processes (Conboy & Morgan, 2011). Thus, information technologies are natural enablers of open innovation. For example, information technologies can make a company's innovation process easily and relatively cheaply transparent. Furthermore, it can enable outsiders to easily observe and take part in these processes.

Yet the organizational innovation literature almost completely ignores the role of information technologies in innovation as can be evident in the lack of information technologies in the summary of 27 years of innovation research, with the exception of idea generation tools (Crossan & Apaydin, 2010). Recently the information systems literature highlighted how and when IT contributes to innovation success (Banker, 2007; Durmusoglu & Barczak, 2011; Kleis, Chwelos, Ramirez, & Cockburn, 2012; Pavlou & Sawy, 2006). Yet, little is known about the role of IT in enabling different innovation in different contexts such as software development (Conboy & Morgan, 2011) and various open innovation settings.

Within the software development context, open source software development communities are prime examples of open innovation as they combine private investment and collective action model of innovation (von Hippel & von Krogh, 2003). Open source software development communities are typically global communities who collaborate through information technologies to develop software whose source code is transparent. At their core, these communities typically have a team of developers. These communities are natural examples of open innovation communities, since the core team systematically draws knowledge from its external community, such as their users. Open innovation communities such as open source software development communities provide competitive advantage to the organizations utilizing them (Hedgebeth, 2007; Lundell, Lings, & Syberfeldt, 2011). The influence of the open innovation provided by the open source software development communities has gone beyond the company level benefits. Open source software altered global competition in the computer software and hardware industries and even in adjacent industries such as consumer goods, where embedded open source software is becoming increasingly common (von Krogh & Spaeth, 2007). The growth of open source software has influenced the society and economy (von Krogh & Spaeth, 2007) and transformed the software industry (Dinkelacker, Garg, Miller, & Nelson, 2002; Wasserman, 2009) by bringing open innovation practices into software development (von Hippel & von Krogh, 2003). All of these changes motivate both the private sector and governments to emulate the open innovation model presented by open source software development teams (Chesbrough, 2003; Davenport, 1997; Goldman & Gabriel, 2005). Large corporations and software companies are incorporating open source software solutions into their product portfolio (von Krogh & Spaeth, 2007), or adopting new software development approaches which utilize open source approaches and technology (Dinkelacker et al., 2002). Governments put in place policies to increase the development and use of open source software (Comino & Manenti, 2005; Cook & Horobin, 2006). Yet, in order to successfully emulate the open innovation practices of open source software communities, companies and governments need to first understand how these communities innovate, and how they use information technologies to enable open innovation.

Yet, as also mentioned earlier, little is done to understand the role of information systems in open innovation setting. Thus, there is a need to adapt the well-known innovation literature to the open innovation settings, which are dependent on information technologies to exist. It is possible to expect that, in the information-technology enabled open innovation setting some of the innovation behaviors from the physical world apply similarly. Yet, it is also likely that some innovation behaviors may manifest differently, while others may not even apply.

Eseryel (2014) attempted to do what is recommended here, by adapting the innovation framework of Nonaka and Takeuchi (1995) for the open innovation setting. This study made important contributions to the literature by developing a content analysis schema that can be used by researchers in analyzing open innovation practices. On the other hand, as also mentioned by the author, the study is limited in its generalizability due to the fact that it is a single case study (Eseryel, 2014). Furthermore, the study is also limited in that is conducted at a single time period despite the fact that both membership and external community participation tend to change over time, which may change the patterns of open innovation (Eseryel, in press). On the one hand, previous research shows that the turnover in membership may be detrimental to effective collaboration, and therefore successful teams should try to both attract new members and retain old members to avoid loss of knowledge and insight (Butler, 2001; Lazar & Preece, 2002; Ma & Agarwal, 2007). On the other hand, some recent research promote the idea that turnover may contribute to an influx of unique contributors, who then increase team innovation through unique ideas, skills and information (Ransbotham & Kane, 2011).

Thus, the goal of this current article is two-fold: One goal is to further validate the open innovation framework (Eseryel, in press) by doing a comparative case study of two open innovation teams. The second goal of this study is to identify how innovation practices evolve over time and how communities use of information systems for innovation, as open innovation communities themselves go through change in membership and leadership. Our research questions can be stated as: How do successful open innovation communities innovate through information technologies? How do the open innovation communities continue to innovate through information technologies as the community changes over time?

LITERATURE

The Challenges of the Extant Innovation Literature for Investigating Open Innovation

Innovation is often described and operationalized in ways that creates confusion on what innovation is and how innovative teams, companies, communities innovate. The confusion about the nature of innovation is evident in the profusion of definitions and operationalizations found in the literature (Garcia & Calantone, 2002). Garcia and Calantone (2002) identified fifteen constructs and at least 51 distinct scale items in only 21 empirical studies on innovativeness.

Furthermore, innovation is typically operationalized by measuring the nature of the technology used or by identifying the market positioning of the product, which constitutes innovation. Innovation literature operationalized its core construct with micro and macro level marketing and technology measures, which disregarded the innovation process. The four types of innovation measures found in the literature are summarized next. Micro-level marketing measures used for empirical analysis of innovation include; the newness of the customers, market approach or competitors from the perspective of the firm (Cooper, 1979), the newness of the product from the firm's perspective (Cooper & de Brentani, 1991), firm's experience of selling the product in a given line of business (Green, Gavin, & Aiman-Smith, 1995), the newness of the product technology to the customer (Ali, Krapfel, & LaBahn, 1995). The micro-level technology measures used for empirical analysis of innovation include the newness of the technology (Goldenberg, Lehmann, & Mazursky, 1999), novelness of the software code (von Hippel & von Krogh, 2003), or the technology knowledge base for the firm (Green et al., 1995) , the newness of the production process for the firm (Cooper, 1979), modification of the technology currently in use at the firm (Colarelli O'Connor, 1998), level of technical difference from a firm's other products (M. Lee & Na, 1994), and the complexity of manufacturing technology (More, 1982). Macro-level marketing measures used to operationalize innovation include its newness to the world

(Atuahene-Gima, 1995), newness to the competitive environment (Cooper & de Brentani, 1991), the consistence of the innovation with existing customer values (Souder & Song, 1997), the lack of actual demand together with the existence of potential demand (Cooper, 1979) and newness to the market (Cooper, 1979). Macro-level technological measures used for operationalizing innovation include the level of science and technology knowledge base within the general scientific community (Green et al., 1995), the level of modification of technology used in other industries (Colarelli O'Connor, 1998) and the extent to which innovation incorporates a substantially different core technology relative to the previous product generation (Chandy & Tellis, 2000).

To sum up, the measures used in empirical studies of innovation keep the innovation process black-boxed. While the outcomes show that innovativeness help companies succeed in the market (Goldenberg et al., 1999), the companies and governments who would like to learn from the literature in order to innovate, do not have a starting point as to which behaviors to encourage, or train for, in order to get their employees to innovate.

The black-boxed nature of innovation operationalization poses challenges particularly for open innovation communities. For example, most recent research measures innovativeness with the number of patents. And yet, in open innovation communities, patenting is the opposite of the norm. Understanding open innovation requires opening the black-box, and understanding what innovating is at it's smallest visible component: a single innovation behavior that one individual contributes to the larger innovation process of an innovation community.

Operationalizing Innovation in the Form of the Human Behavior "Innovating"

Nonaka and Takeuchi's (1995) work on innovation imparts knowledge from the Japanese companies' innovation processes. This work identifies specific behaviors for innovation, which enables emulation by other companies. This research finds its core at the knowledge management literature: Nonaka and Takeuchi (1995) identify knowledge creation as the source of continuous innovation. Innovative Japanese companies gain their innovative status by continuously creating knowledge to continuously innovate, possibly in incremental ways. Japanese companies do this by incorporating external knowledge into their company, disseminating this external knowledge internally, incorporating it into their products and technologies, and sharing their newly created knowledge back with the external world. Thus, despite the fact that in the information systems literature innovation and knowledge management are treated as separate literature streams, we suggest that they be seen as the same literature stream as at the core of the innovation lies knowledge creation, transfer and management (Eseryel, 2014).

Innovating is a part of software development process, since innovating is the means to solve critical software problems (Brooks, 1995). Innovating plays a role in various stages of software development from requirements analysis through program design (Cougar, 1990). Each of these stages requires very specialized knowledge. Furthermore, creating new knowledge (i.e. innovating) is a 'must' in software development because technology changes very rapidly. The complexity of the solutions needed for software development are increasing (Cougar, 1990). Thus, knowledge is a critical resource for software development (Robillard, 1999) and knowledge creation is at the core of innovation, also within software development.

Creating a successful innovative open innovation model requires engaging external members and utilizing their knowledge (von Krogh, Spaeth, & Lakhani, 2003; West & O'Mahony, 2005). A first step to emulating innovative open source software development teams is understanding how knowledge creation happens in these settings, and how various participants contribute to this continuous innovation (continuous knowledge creation) process (Agerfalk & Fitzgerald, 2008). While at the heart of innovation is knowledge creation, the enabler of knowledge creation in open innovation communities is information technologies. This is especially the case for open source software development communities since all of the work and related communication is done through information technologies. This provides nearly unlimited transparency for the users and other stakeholders of the software and the software development project. Yet, the exact role of how information technologies facilitate open innovation in software development is not known (Conboy & Morgan, 2011). A key research gap in this area is whether and how information technologies facilitate effective contributions from developers and users, which may contribute to increased innovation (Setia, Rajogopalan, & Sambamurthy, 2012). Yet the organizational innovation literature almost completely ignores the role of information and communication technologies in innovation as can be evident in the lack of information technologies in the summary of 27 years of innovation research (Crossan & Apaydin, 2010). Crossan and Apaydin's (2010) review of the innovation literature shows that the only instances where the information technologies were considered as an important part of innovation model was related to the use of ideation technologies during the ideation process of innovation (Cebon & Newton, 1999; Losh, Stein, & Terwiesch, 1996). The next section summarizes the extant research on open innovation and knowledge creation by the open source software development teams.

Open Innovation by Open Source Software Development Communities

This section first introduces the general structure of open source software communities (although no two communities are exactly the same). We then describe how these communities innovate according to the extant literature.

The open source software communities have at their core a team of developers. In their periphery, they have active and passive users, and thus their structure can be likened to an onion (Crowston & Howison, 2006). While it is likely for the individuals residing at the inside layers of the onion structure to contribute more to innovation, the research strongly emphasizes the importance of the peripheral members. Thus, it is difficult to know the real source of knowledge contribution within the innovation process.

While the open source software core developers actively design and develop the software, the users contribute to software development by generating patches and reporting bugs (G. K. Lee & Cole, 2003) and sharing their knowledge (AlMarzouq, Zheng, Rong, & Grover, 2005; Cox, 1998; Crowston & Howison, 2005; Gacek, Lawrie, & Arief, 2001; Mockus, Fielding, & Herbsleb, 2002). Peripheral users contribute to increased popularity of the software by the sheer act of using the software (Setia et al., 2012), to increased product quality by putting in their time (Setia et al., 2012) and to open innovation by providing novel ideas (Chesbrough, 2003; Setia et al., 2012). The cost of sharing innovation in the form of novel software code is minimal for the developers and users (von Hippel & von Krogh, 2003), while the benefits may be high in terms of reputation and reciprocity (Lerner & Triole, 2000; von Krogh, 2002). This dynamic was clearly observed in Linux, a well-regarded innovative open source software development community: There, the users generated patches and reported the bugs (G. K. Lee & Cole, 2003), thus contributing their knowledge to the innovation process. The developers tested user patches, and suggested necessary changes for user patches (G. K. Lee & Cole, 2003). By sharing unique ideas, users contributed both to software quality and innovation (G. K. Lee & Cole, 2003).

The dynamic among the open source software development community members described above indicate that open source software development communities are open innovation communities which utilize private-collective innovation model (von Hippel & von Krogh, 2003). It is a private innovation model because individual participants use their own time and resources to innovate. At the same time, it is a collective innovation model, since many developers and users collaborate to innovate by exchanging knowledge and building upon each others' knowledge and work. Differently than proprietary innovation models, developers create most of the innovation (von Hippel & von Krogh, 2003).

The knowledge creation within the open source software teams manifest through a plethora of learning opportunities (Hars & Ou, 2001; Hermann, Hertel, & Niedner, 2000; Himanen, Torvalds, & Castells, 2001; Kohanski, 1998) and socialization practices (Crowston & Annabi, 2005; Weber, 2004). This learning, which is the first step of knowledge creation, begins at open source software development communities through feedback and error correction (Kogut, 2000; G. K. Lee & Cole, 2003), which is identified as an important part of innovating communities in addition to critical evaluation of knowledge (G. K. Lee & Cole, 2003). These behaviors manifest themselves in open source software development communities in the form of developers reviewing the source code that is submitted by others, and giving feedback to the source code and/or fixing it when necessary (G. K. Lee & Cole, 2003). There is a peer-review process that is present in these open innovation communities (G. K. Lee & Cole, 2003). This peer-review process is continuously present due to the transparencies enabled by these communities' dependence on information technologies as the sole medium of communication, coordination and work. The information technologies that are archived and publicly available allow the developers to rapidly exchange knowledge, compare facts, search, and discuss any change (G. K. Lee & Cole, 2003). This way, the archived knowledge also extends the transparencies beyond synchronous interactions. The stakeholders can go back in time and experience the same knowledge transfer, as if the knowledge creation is happening a the moment they read the mailing lists. This results in a continuous cycle of knowledge growth.

The Framework to Identify Open-Innovation Behaviors at Open Source Software Development Communities

Nonaka and Takeuchi (1995) identified four modes of knowledge creation through their investigation of highly innovative Japanese companies and what the company employees do to be able to continuously innovate. These four modes of knowledge creation and how they relate to the open innovation context of open source software development teams are described next.

Socialization. Nonaka and Takeuchi (1995) define socialization as transfer of tacit knowledge among people through an apprenticeship process or a similar experience. Tacit technical skills are transferred through observation, imitation and practice. An apprenticeship model typically refers to the on-the-job learning found in brick-and-mortar settings. Yet, open source software development communities have their own version of apprenticeship model, which begins with testing and reporting bugs in the software, followed by submitting patches to fix the identified bugs (Ducheneaut, 2005; von Krogh et al., 2003). While the software code is explicit knowledge (Haefliger, von Krogh, & Spaeth, 2008; Morner & von Krogh, 2009), "the way it is structured and built conveys more tacit knowledge" (Morner & von Krogh, 2009, p. 443). This can be better understood by viewing explicit and tacit knowledge as mutually complementary entities on the same continuum rather than being distinct opposites (Jha, 2002; Nonaka & von Krogh, 2009).

Secondarily, the tacit knowledge is transferred through the observation of the community-based decision-making process that goes into the artifact. Transferring others' tacit knowledge through the source code artifact, and through participating in online decision-making regarding the software code happens through an intense intellectual engagement with the artifact and the underlying reasoning and choices (Eseryel, 2014), by examining for example, which algorithms others use and how they structure the source code (Morner & von Krogh, 2009). This intellectual examination causes the software code (explicit knowledge) to lose its explicitness and increases its tacitness (Nonaka & von Krogh, 2009) in the minds of those who interact with the software artifact. Eseryel (2014) identified three distinct socialization behaviors for the IT-enabled context of open source software development communities. These include (1) testing and new feature ideation, which refer to typically the socialization behaviors of the new participants, although it is not limited to newcomers; (2) software development (novel patch development); and (3) action-embedded contribution to others' software development, which refers to testing others' patch submissions, improving them as needed and committing (adding) these to the software code (Eseryel, 2014).

Externalization. Externalization is a process of converting tacit knowledge into explicit concepts through the act of writing, dialoging and collective reflection (Nonaka & Takeuchi, 1995). In the brick-and-mortar setting, externalization happened often through the use of metaphors and analogies, which created shared understandings. Eseryel (2014) did not find evidence for the use of metaphors or analogies in the information-technology based communication and collaboration data of the open innovation community. Instead, researchers observed externalization of tacit knowledge in problem conceptualization (Hemetsberger & Reinhardt, 2006), new idea creation (Hemetsberger & Reinhardt, 2006) and problem resolution (Eseryel, 2014) within open source software development communities. The knowledge externalization happened often in mailing lists, as individuals explained their ideas, evaluated, rejected, corrected or defended certain ideas (Hemetsberger & Reinhardt, 2006). On the one hand, knowledge externalization manifested differently in IT-enabled open innovation communities than the settings investigated by Nonaka and Takeuchi (1995). On the other hand, the knowledge externalization behaviors discussed above meet the requirements of Nonaka and Takeuchi's (1995) externalization knowledge creation mode, which refers to creating shared understandings, dialogue and collective action by explicating tacit knowledge (Eseryel, in press). Eseryel (2014) identified five ways in which software developers externalize their tacit knowledge as part of the problem solution process: (1) knowledge contribution to help others resolve a problem, (2) solution creation for problems, (3) knowledge dissemination about the detailed logic of their recommended solutions, (4) brainstorming potential solutions, their pro's, con's, etc. and finally (5) mentoring and guiding others about how to best contribute to the software development process.

Combination. Combination refers to reconfiguring explicit information through sorting, adding, combining and categorizing it, which may lead to the creation of new knowledge (Nonaka and Takeuchi, 1995). Nonaka and Takeuchi (1995) include the knowledge exchange through documents, meetings, and conversations as combination activities. The combination activities create systemic knowledge: The combination of knowledge in central repositories then enable new knowledge creation, for example by data mining, and business intelligence techniques, which support further decision making. Two types of knowledge combination were identified in open source software development communities' archival data (Eseryel, 2014); (1) knowledge dissemination about issue resolution, or software development in a central location, which can then be datamined, and (2) combination of external knowledge sources, by referring or linking to other knowledge source from within a knowledge source.

Internalization. This knowledge creation mode refers to converting explicit knowledge into tacit knowledge through "learning by doing" (Nonaka & Takeuchi, 1995). The main means of internalization is documentation, i.e. explicating and diagramming knowledge into documents and manuals (Nonaka & Takeuchi, 1995). Within software engineering, knowledge construction includes testing and documentation (Kautz & Thaysen, 2001). Testing and documentation are such important ways of internalizing the software development knowledge that their integration into the computer science and software engineering curricula is recommended (Jansen & Saiedian, 2006). Software documentation requires intellectual engagement with the system, such as by experimentation with the system to learn more about it (Eseryel, 2014). On the one hand, the documentation process results with an explicit artifact, in the form of documents. These documents "facilitate the transfer of explicit knowledge to other people, thereby, helping them experience the experiences of others indirectly (i.e., 're-experience' them)" (Nonaka & Takeuchi, 1995, p. 69). On the other hand, the act of documenting also results in an increase of tacit knowledge in the mind of the documenter, in the form of meaning making. Four types of internalization were identified by prior literature that applies to the open source software development communities: (1) software documentation, and (2) test case development, which refer to the internalization of technical knowledge, and (3) project related web-page development, and (4) wiki contributions, which refer to the internalization of project-based knowledge as well as software based knowledge (Eseryel, 2014).

RESEARCH METHOD

This section introduces the context of the study, provides an overview of the study in its whole, describes the case selection, archival data collection and reduction, and the analysis of the data.

Context of the Study

Not all open source software development communities are the same. Open source software development communities within the Apache Software Foundation were chosen for this study because Apache provides a revelatory setting for innovation: Apache setting provides

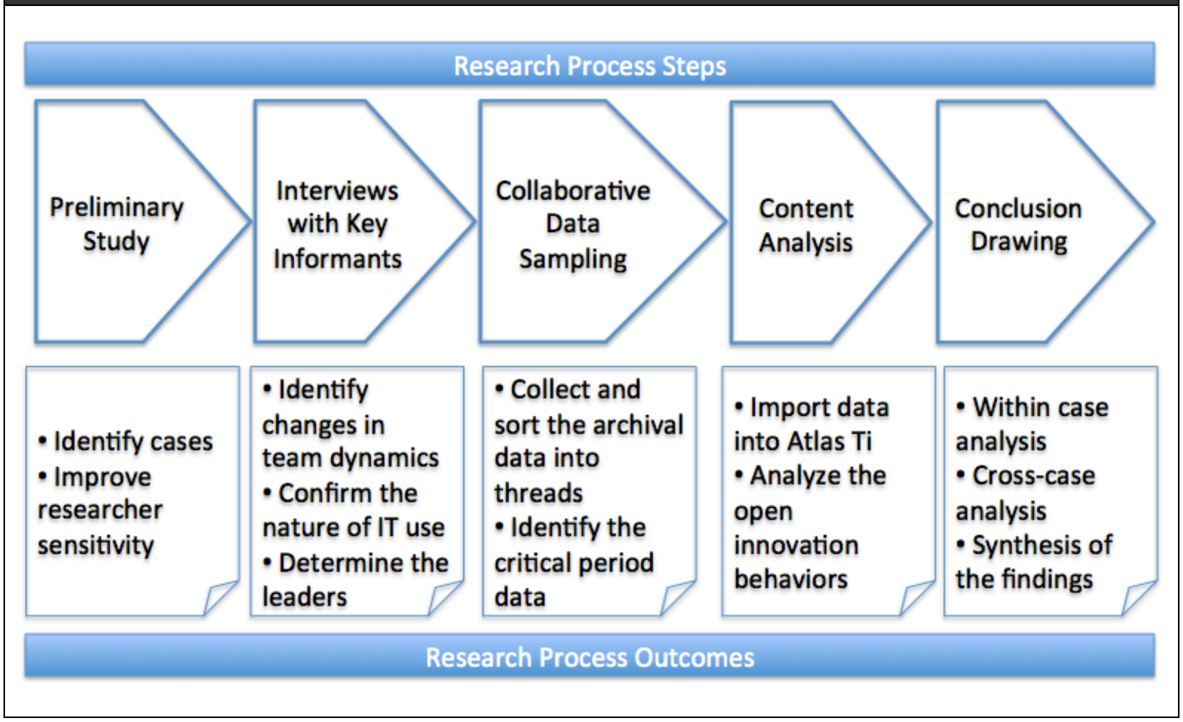
Journal of Leadership and Management®
An International Journal of Strategic, Behavioral and Public Aspects of Leadership in Management
Cabell's Listed
A journal of the Institute of Leadership and Management Association, PO Box 564, Douglassville PA 19518. 484-332-3331. https://www.jleadershipmanagement.org

one of the few extremely successful models of open innovation and open source software development (Thomas & Hunt, 2004). Apache Software Foundation has managed to transfer the successful open innovation practices of the Apache web server team (Fielding, 1999) to more than a hundred other open source software development teams that reside under the Apache Software foundation. Understanding these open innovation practices are important for organizations for emulation purposes. Many developers at Apache at the same time used the software that they developed within these projects for their work at their companies. This made the Apache setting an interesting one for companies that plan on participating in/adopting existing open source projects. Moreover this setting created a learning opportunity for organizations that plan on developing a similar information-technology enabled open innovation process. In terms of governance, Apache open source software communities can also be thought of as voluntary teams (with an extended community outside of the team, from which the team receives its members as well as external contributions). These teams resided under a non-profit organization called Apache Foundation (Bonaccorsi & Rossi, 2006; Wasserman, 2009). Thus, similar to organizational self-managing teams, Apache teams had an umbrella organization with organizational structures, information technology infrastructure, norms and reporting requirements (Ljungberg, 2000). Although this umbrella organization influenced Apache projects, each team independently determined its operations (The Apache Software Foundation, 2012).

Overview of the Study

This investigation started off with a preliminary study (interviews with Apache Software Foundation members) to understand the context and to select the right cases, which was crucial for the success of the study. This is followed with the 2 rounds of interviews with key informants of each of the selected projects to identify the changes in community dynamics, determine the leaders of the project and to confirm the project members' views on the innovativeness of the project and how they collaborated together using information technologies. Each interview round was followed with content analysis of the archival data preceding the interviews to observe the nature of open innovation within the open innovation community during the time period that represents the interviews. The archival data for this study was sampled with input from the community members. Lastly, conclusions were drawn from the two case studies using within- and cross-case analyses. The findings were synthesized and compared with the extant literature in order to systematically discuss the findings. Figure 1 below presents the key steps and outcomes of this research.

Figure 1. The Research Process Steps & Outcomes



Case Selection

Pettigrew (1990) recommended doing low-cost preliminary studies for a better understanding of the frameset before selecting specific cases for longitudinal studies requiring intensive access. Thus, a preliminary study was conducted via interviews with long-term Apache Software Foundation members (1) to inform the researcher about the context, (2) to improve the researcher sensitivity, i.e., "ability of the researcher to pick up subtle nuances in the data that infer to a meaning" (Corbin & Strauss, 2008, p. 19) and (3) to select the appropriate cases for this longitudinal study (Pettigrew, 1990). While the general findings of the preliminary study are out of the scope of this study, the study helped identify the two very successful examples of open innovation to be investigated. To identify two very innovative and successful projects, formal and informal interviews were conducted with Apache Software Foundation Board members, who followed all of the Apache projects and actively participated in a number of these projects.

During the preliminary study, several communities were identified as potential cases, which were expected to stay relatively stable at least in the short term, and others that were more dynamic in their membership and expected to show changes over time. This current study investigated one Apache community, which stayed relatively stable in terms of community membership and leadership (referred to as "Stable" from here on), and another with major changes in the community membership and leadership (Referred to as "Delta" from here on). These two cases were identified to allow for theoretical replication (Guba & Lincoln, 1994). In theoretical replication, two cases are expected to show different results for predictable reasons.

In order to control for other factors that may potentially influence open innovation, two projects were selected to show similarities. Both projects were modular in nature, and both projects satisfied the minimum size requirement for studying teams by having seven or more members (Hare, 1976): In the first time period, Delta had 11 core team members Stable had 12. Both teams also had a vibrant community. Delta had 129 users and Stable had 45 users who contributed to the project as part of their extended community of the project. Moreover, both teams were at the development stage (Helfat & Peteraf, 2003), as opposed to the founding or maturity stages. This team stage indicated stability in these two communities' membership due to existence of clear norms and roles. Each project had a stable version of the software and a common goal. Software development projects at the development stage presented an active and dynamic software development environment, with much novel software development activity, communication, and many open innovation activities, as defined by von Hippel and von Krogh (2003).

Introduction of the Archival Data

Archival data were used to objectively observe the knowledge creation behaviors by community members. Open source software development communities depend exclusively on information technologies for their communication, task coordination, decision-making and work. Thus, the freely available archival data provide a near-complete archive of the open innovation behaviors of all team members. One clear exception to this is the bi-annual conference meetings, which provide a face-to-face interaction opportunity to community members. Even then, the members are often observed sitting around a table, doing work on their laptops and communicating with each other through mailing lists instead of talking to each other (Crowston, Howison, Masango, & Eseryel, 2005). They reported doing this to include the community members who were not at the conference, and to abide by the rule that all key activities and important decisions must be made on information technologies determined for the software development and related collaboration. Therefore, the archival data enable an objective analysis of open innovation activities. At least a part of the open source software development community members (core team members) also share connections on social media such as Facebook, twitter and LinkedIn. The researcher became connected to the core team members on social media to observe whether open innovation activities may at least partially be conducted there. However, it is observed that social media such as Facebook is often used for sharing personal news rather than a project-based interaction expected to be observed for creation of open innovation.

In this study, four types of information technology artifacts were analyzed for evidence of open innovation.

Two of the analyzed IT artifacts were communication media: Developers' mailing list, captured core-team based knowledge creation and users' mailing list helped identify the inflow and outflow of knowledge between the core team and the extended community required by open innovation. These mailing lists were organized by thread per the subject being discussed. Thus, they enabled the observation by all community members of all relevant issues that were raised, and decisions that were made related to each subject.

The third IT-artifact was the issue tracker where the quality improvement efforts and ideation, which resided at the core of open innovation (Setia et al., 2012) were coordinated within the whole community. Both communities that were investigated in this study used the same issue tracker: JIRA. Each JIRA issue stored the following information fields: the type of software issue or problem, its URL, the project name, the version of the software and the relevant module/feature/component, issue creator, the person assigned to the task, issue status (open, resolved, etc.) and the issue priority. Both Stable and Delta set the JIRA to forwarded new entries to the developer's mailing list via email, using the subject line of the issue as the email subject line. For this study, the JIRA messages on the same subject were organized in threads, for coding these issues in the same way mailing list threads were coded.

The last IT-artifact that is analyzed is the software version control tool (SVN), which was the system that enabled the developers to simultaneously work on software development and submit their changed codes into the software code (also referred to as committing code). SVN allowed the researcher to identify members' software code contribution behaviors, a form of knowledge contribution (Morner & von Krogh, 2009).

Both SVN and JIRA provided systematic information on various aspects of the open innovation process, such as ideation (Setia et al., 2012), quality improvement (Setia et al., 2012) and action-embedded innovation (Eseryel & Eseryel, 2013).

Archival Data Collection and Reduction: Sampling of Threads.

All data were organized in the form of threads based on their subject line. Single-email threads were eliminated, as they did not present opportunities for community based open innovation. To capture change over time, threads were sampled at two time points: Time 1 indicated the period that the projects were identified by the Apache Software Foundation

Journal of Leadership and Management®
An International Journal of Strategic, Behavioral and Public Aspects of Leadership in Management
Cabell's Listed
A journal of the Institute of Leadership and Management Association, PO Box 564, Douglassville PA 19518. 484-332-3331. https://www.jleadershipmanagement.org

board members as successful open innovation projects. Time 2 indicated the time period shortly after a change in the project. At the Delta project, a major change occurred in the form of an inflow of a number of new and highly active members, as well as an ongoing reduction in activity and leadership provided by the founder of the project. At the Stable project, a minor change occurred in the form of the founder of the project handing his or her formal "chair" title over to another person, while still continuing to provide strong leadership.

Eseryel (2014) identified two types of contributions as relevant for open innovation: ongoing regular contributions and critical contributions. Individuals' ongoing regular contributions contributed to incremental innovation. Secondly, people's contributions to the critical events (such as by fixing a major bug that blocks the release of a software, or adding a feature that is deemed to be highly important to the success of the project) were also key to the success of the open innovation project. Developers' versus users' open innovation patterns between regular versus critical events differed (Eseryel, 2014).

Table 1. Archival Data Analyzed for Both Teams

	Threads Just Before the 1st Interview	**Critical Event Threads Before the 1st Interview**	**Threads Just Before the 2nd Interview**	**Critical Event Threads Before the 2nd Interview**	**Total Threads**
THE STABLE PROJECT (Total of 1015 Messages Analyzed)					
SVN Messages (Work)	20	43	20	38	121
JIRA Threads	20	16	20	28	84
Developers' Mailing List Threads	20	5	20	12	57
Users' Mailing List Threads	20	1	20	0	41
TOTAL	**80**	**65**	**80**	**78**	**303 Threads (1015 Messages)**

THE DELTA PROJECT (Total of 1193 Messages Analyzed)					
SVN Messages	20	17	20	21	78
JIRA Threads	20	12	20	18	70
Developers' Mailing List Threads	20	3	20	1	44
Users' Mailing List Threads	20	7	20	5	52
TOTAL	**80**	**39**	**80**	**45**	**244 Threads (1193 Messages)**

Eseryel (2014) also found that when sampling 20 threads or fewer per archival data type, it was possible to reach a theoretical saturation where adding new threads did not contribute further to the theory (Morse, 2004). Therefore, the same sampling strategy (Eseryel, 2014) was replicated and adapted to a longitudinal investigation on how innovation process unfolds over time. This resulted in a sampling of a total of 320 regular-event threads from both projects and matching critical event threads (227 threads) that are found within the same time period. This sampling strategy resulted in the analysis of a total of 547 threads (2208 messages). These threads represented 1015 messages from the Stable project and 1193 messages from the Delta project. Table 1 below shows the distribution of these threads across two projects.

Critical contributions were identified first by filtering the (JIRA) issue trackers for ‘major’ issues (an important problem, which still allowed the users to use the software), ‘critical’ issues (an important issue which allowed a minor release, but not a major one) or ‘blocker’ issues (an issue that blocked all releases). The selection of significant events was limited to the six months preceding the points in time that were identified for the study. This way, the behaviors captured with the archival data matched the nature of the project as reflected in the interviews. The critical issues were then reviewed and shortlisted by a key informant member of each of the development teams. Once the critical contributions from the SVN were shortlisted, the technical terms and the issue numbers were used to identify all relevant discussions on the developers’ and users’ mailing lists. The resulting number of threads was provided in Table 1.

Data Analysis

a) Interview Analysis. The interviews were mainly used to (1) identify the community leaders and leadership change, and (2) to confirm whether the expected changes in the community dynamics happened or not. The interviews were conducted with key informants of the team who were present at Apache conferences. All interviews were transcribed and then analyzed inductively by two independent content analyzers using Atlas-Ti content analysis software. Calculation of whether the leadership of the projects changed or not happened through identifying perceived leaders at each time point. Perceived leaders were identified by calculating perceived leadership indices (PLI) for each member at each interview period (Sarker, Grewal, & Sarker, 2002). PLI was calculated for each participant by dividing the number of times that individual was identified as a leader by members of the team by the total number of interviewees (Sarker et al., 2002). All members' perceived leadership index scores ranged between zero (non-leader) and one (perceived as a leader by all interviewees). Following Heckman and Misiolek (2005), members with higher than or equal to 0.5 PLI were identified as strongly accepted leaders, in that at least half of the group members or more perceived them as leaders.

b) Content Analysis of the Archival Data. The data was organized in threads and content analysis was done through Atlas-Ti software. The content analysis was conducted at the thematic level. At thematic level coding, one may select a whole message or any sub part of the message as the unit of coding as long as the unit captures a meaning.

To capture the knowledge creation behaviors that reside at the core of community based open innovation process, the knowledge creation framework of Nonaka and Takeuchi (1995) was utilized. This framework was developed through an investigation of highly innovative Japanese companies and what the company employees do to be able to continuously innovate. The framework was especially useful in capturing innovation behaviors, since the framework captured what individuals did as part of innovating, rather than trying to measure innovation as an outcome or output. In its current form, the knowledge creation framework was relevant to face-to-face innovation process that is found within brick-and-mortar organizations. This framework, in its current form was found difficult to apply to the information-technology dependent open innovation communities. The knowledge creation modes that made up the framework were generally described by Nonaka and Takeuchi 1995) with various examples from brick-and-mortar organizations. Eseryel (2014) adapted this framework for the IT-enabled open innovation setting of open source software development communities. This adaptation also resulted in a content analysis schema development for capturing open innovation behaviors at open source software communities (Eseryel, 2014). This schema was used in this study and the archival data was deductively analyzed using this schema. Detailed justifications on how the adaptation of each knowledge creation behavior fits with the original framework are beyond the scope of this article, as they were described in detail by Eseryel (2014).

This previous study had established the reliability of the content analysis schema at 87% (Eseryel, 2014), which is considered high level of reliability (Neuendorf, 2002). For the current study, two analysts independently content analyzed 20 episodes at a time per project and resolved all the differences in coding through discussion. Two coders independently analyzed a subset of the data to establish content analysis schema reliability for this study. The participation of two content analysts in reliability establishment enabled the identification and elimination of mistakes. In general, reliability test sample size can make up 10% or more of the data (Potter & Levine-Donnerstein, 1999), yet, an acceptable level of reliability can always be reached with fewer than 300 episodes (Neuendorf, 2002). After the second round of coding, the reliability of the content analysis schema for this study surpassed the original rate of 87%. After this, the coding was completed by one analyst using the reliable content analysis schema, as it is a commonly accepted practice for researchers to code their complete dataset (e.g., Lapointe & Rivard, 2005; Levina & Vaast, 2005; Pawlowski & Robey, 2004). As data analysis is at the heart of building theory from case studies (Eisenhardt, 1989), it is best done by the researcher (Eseryel, 2014).

The results of the analyzed data were then displayed by transferring the coding results into Microsoft Excel and tabulating different findings for easier comparison. The findings were then synthesized first by conducting within case studies and then cross-case comparisons (Eisenhardt, 1989). The analysis process used here adopted the steps of data analysis as recommended by Miles and Huberman (1994).

FINDINGS

This section presents the answers to the two research questions. To answer the first research question, namely "How do successful open innovation communities innovate through information technologies?", we provide a baseline of the both projects in the first time period, when we expected them both to be relatively stable. Shortly before the first time period, new members had joined both projects. Next, the findings from Stable and Delta are provided in order to exhibit how the open innovation community dynamics change as the community changes over time through a turnover in leadership and a change in membership.

Open Innovation Behaviors Under Relatively Stable Community Conditions

The coding verified the observance of all codes that were determined by the previous research (Eseryel, 2014) for the given two case studies (Table 2). While the data was also scanned for potentially new inductive codes, no new codes emerged.

The coding identified 741 instances of knowledge creation behaviors between the two projects during the first time period. Table 2 presents the innovation behaviors and the relevant knowledge creation modes and behaviors. The four knowledge creation modes were categorized into three open innovation processes. The first process was the ***action-embedded open innovation process***, where individuals innovated by contributing to the work of software development (Eseryel, 2013). The outcome of their work then became boundary objects for community members to transfer their tacit knowledge. The second process was called the ***integral open innovation process***, where individuals innovated together through intense interaction and knowledge exchange with internal and external community members. And the last open innovation process was called peripheral open innovation process. The process was called "peripheral” because innovating in this manner involved the uni-directional communication related to action-embedded innovation, as well as the highly peripheral work that supported the community's goal (of high quality software development) at a more superficial level.

Among the fifteen types of knowledge creation behaviors, three socialization behaviors made up the action-embedded open innovation process, 5 externalization behaviors made up the integral open innovation process, and 3 combination behaviors together with 4 internalization behaviors made up the peripheral open innovation process.

Journal of Leadership and Management®
An International Journal of Strategic, Behavioral and Public Aspects of Leadership in Management
Cabell's Listed
A journal of the Institute of Leadership and Management Association, PO Box 564, Douglassville PA 19518. 484-332-3331. https://www.jleadershipmanagement.org

Table 2. Identified Open Innovation Processes and the Relevant Knowledge Creation / Innovation Behaviors

Open Innovation Process	**Knowledge Creation Mode** (Nonaka & Takeuchi, 1995)	**Knowledge Creation Behavior (Eseryel, 2014)**
Action-Embedded Innovation	Socialization *(Tacit to Tacit Knowledge Creation)*	New Feature Ideation and Testing
		Software development (Novel Patch Development)
		Action-embedded contribution to others' software development
Integral Innovation	Externalization *(Tacit to Explicit Knowledge Creation)*	Knowledge contribution for problem resolution
		Solution creation for software development problems
		Knowledge dissemination about recommended solutions
		Brainstorming potential solutions
		Mentoring and guiding others
Peripheral Innovation	Combination *(Explicit to Explicit Knowledge Creation)*	Knowledge dissemination on issue resolution, which may be distributed across various ITs
		Knowledge dissemination on software development acts (patch commit) in a central location
		Combination of external knowledge sources (from discussions, project webpage, wiki or external knowledge)
	Internalization *(Explicit to Tacit Knowledge Creation)*	Documentation, thereby internalization of software-related knowledge
		Test case development thereby internalizing software-related knowledge
		Project Related Web-page development, thereby internalizing project-based knowledge
		Wiki contribution, thereby internalizing project-based or software-based knowledge

Journal of Leadership and Management®
An International Journal of Strategic, Behavioral and Public Aspects of Leadership in Management
Cabell's Listed
A journal of the Institute of Leadership and Management Association, PO Box 564, Douglassville PA 19518. 484-332-3331. https://www.jleadershipmanagement.org

Table 3 presents the three open innovation processes and the frequency of their use based on the archival data analysis. Next, we discuss these three open innovation processes and the relevant knowledge creation modes and behaviors.

Table 3 Four Knowledge Creation Modes and the Frequency of Their Use

Open Innovation Process	The Stable Project Time 1	The Stable Project Open Innovation Process Use Frequency	The Delta Project Time 1	The Delta Project Open Innovation Process Use Frequency
Action-Embedded	123	36%	94	24%
Integral	163	48%	227	57%
Peripheral	55	16%	79	20%
Total	341		400	

Action-Embedded Open Innovation Process

Action-embedded open innovation process was the second most frequently used open innovation process with respectively 36% and 24% of the knowledge creation behaviors in Stable and Delta projects falling into this category. This category had originally consisted of the socialization behaviors, which were defined as transferring tacit knowledge through dialogue, observation, imitation or guidance (Nonaka and Takeuchi, 1995).

At the core of action-embedded open innovation process was socialization through learning by doing, which mainly consisted of contribution to software development within this context. Development of novel software was one of the means of knowledge creation for the team members. Another means was, contribution to others' software development, through testing, improving and committing (adding) to the software core. For example, individuals who read the following ideas for improving the implementation of a feature learned about the correct implementation of the spans interface [general software development knowledge], and how an improper implementation of the spans class may influence various features of the software being developed [specific knowledge related to the current project].

> *"In TermSpans (or the anonymous Spans class returned by SpansTermQuery, depending on the version), the skipTo() method is improperly implemented if the target doc is less than or equal to the current doc: public boolean skipTo(int target) throws IOException {*

> *//are we already at the correct position?*
> *if (doc >= target) {*
> *return true;*
> *}*
> *...*
> *This violates the correct behavior (as described in the Spans interface documentation), that skipTo() should always move forwards, in other words the correct implementation would be:*
> *if (doc >= target) {*
> *return next();*
> *}*
> *This bug causes particular problems if one wants to use the payloads feature - this is because if one loads a payload, then performs a skipTo() to the same document, then tries to load the "next" payload, the spans hasn't changed position and it attempts to load the same payload again (which is an error)."*

This example exemplifies how the action-embedded innovation behaviors transferred knowledge about software development in general. Furthermore, these behaviors helped socialize new members, by showing them the inner-workings of the software and the general process by which the community members collaborated. One interviewee, when asked how he learned more about the content of the project mentioned:

> *"This is what you should do, start by testing, reporting bugs, and then submit small patches, download and analyze others' work and lurk into what is going on on the list. This is how you are expected to learn about the project. Nobody will explain you all they know, or give you tasks to do."*

This process created sympathized knowledge according to Nonaka and Takeuchi (1995), in the form of shared mental models. The interviewees also referred to how they started to share others' view of the software as a result of inspecting others' code, another key element of this study. One interviewee mentioned how he learned from a key developer by inspecting his/her work.

> *"You look at the code s/he writes, and you really admire what s/he does. I would like to be like him. So I try to replicate his/her thought process sometimes."*

This quote provides a representative example of how individuals transferred tacit knowledge from each other through intellectual engagement with the software artifact that they create. Yet, this knowledge transfer did not happen through the physical observation of the act of software development, as it would be expected from a brick-and-mortar apprenticeship model. Rather, the transfer of tacit knowledge happened primarily through inspecting the final output of the software development act, namely the software artifact.

Yet, this was not the only outcome of the socialization behaviors. The form of socialization used for the open source context also referred to the core operations, and the main tasks to achieve the goal of this group, namely, software development.

The socialization that happened in this open innovation context is one where the main interaction is with the information technology artifact to directly produce an information technology outcome.

Integral Open Innovation Process

This process was the most frequently used open innovation process observed in the data: 48% of the Stable Project innovations and 57% of the Delta Project innovations followed this process. This showed that the most frequently used knowledge creation form involved tacit to explicit knowledge transfer.

In this form of innovation the role of information technologies was to create interaction among people, which then resulted in a community-based innovation process. While the conversion of tacit knowledge to explicit knowledge (i.e., externalization) is known to be, by definition, very challenging, if not impossible, the archival data analysis indicated that the developers did it very often. In fact, the interviews indicated that the developers were conscious about how they externalized their knowledge in order to get the community buy in. As one interviewee put it:

> *"When you put forth an idea that you would like to get implemented, you need to communicate it clearly so that everyone knows where you're coming from. That's why I try to be very clear about my communication on the [mailing] lists. I always explain what the alternatives are, why my solution is the best one, and how I will go about it and why. Doing so, you make sure that everyone gets behind the idea faster, so that you can go forward with it."*

The fact that individuals are consciously externalizing their knowledge, in an effort to get buy-in from others regarding their recommended solution was also often reflected in their emails. For example, after giving an extensive explanation and evaluation for his/her recommendation, one developer asks:

> *"What do you think? We can do this change in [version] 3.0 so that we don't have to take care of backward compatibility issues, that is of course if everybody agree[s] to make the change."*

Externalization manifested mostly as part of the problem identification, definition and resolution, which are integral steps of software development. The example below presents an instance, where a developer presented why the "company rule" did not work correctly, why this was a problem and identified two potential solutions:

> *" The COMPANY rule in StandardTokenizer is defined like this:*
> *// Company names like AT&T and Excite@Home.*
> *COMPANY = (ALPHA) t ("&"|"@") (ALPHA)*
> *While this works perfect for AT&T and Excite@Home, it doesn't work well for strings like widget&javascript&html.* ***[PROBLEM DEFINITION]*** *Now, the latter is obviously wrongly typed, and should have been separated by spaces, but that's what a user typed in a document, and now we need to treat it right....*
> *That got me thinking on whether this rule is properly defined, and what's the purpose of it. Obviously it's an attempt to not break legal company names on "&" and "@", but I'm not sure it covers all company name formats. For example, AT&T can be written as "AT & T" (with spaces) and I've also seen cases where it's written as ATT....* ***[PROBLEM DEFINITION]***
>
> *This rule slows StandardTokenizer's tokenization time, and eventually does not produce consistent results.* ***[IDENTIFICATION OF THE NEGATIVE OUTCOME OF THE IDENTIFIED PROBLEM]*** *If we think it's important to detect these tokens, then let's at least make it consistent by either* ***[SOLUTION CREATION FOR SOFTWARE DEVELOPMENT PROBLEMS****]:*
>
> *- changing the rule to (ALPHA)(("&"|"@") (ALPHA))+, thereby recognizing "AT&T", and "widget&javascript&html" as COMPANY. That at least will allow developers to put a CompanyTokenFilter (for example) after the tokenizer to break on "&" and "@" whenever there are more than 2 parts. We could also modify StandardFilter (which already handles ACRONYM) to handle COMPANY that way.* ***[KNOWLEDGE DISSEMINATION ABOUT THE RECOMMENDED SOLUTIONS]***
>
> *- changing the rule to (ALPHA)("&"|"@") (ALPHA)(*P* | "!" | "?") so that we recognize company names only if the pattern is followed by a space, dot,*

> *dash, underscore, exclamation mark or question mark. That'll still recognize AT&T, but won't recognize widget&javascript&html as COMPANY (which is good).* ***[KNOWLEDGE DISSEMINATION ABOUT THE RECOMMENDED SOLUTIONS]***
>
> *What do you think?* ***[EFFORT TO BRAINSTORM POTENTIAL SOLUTIONS]"***

Often the problem resolution seemed to be complex with many dependencies to consider. Similar to the example above, often several ways of resolving an issue were discussed and their pro's and con's were determined in an effort to find the most effective and efficient way of resolving the problem. Others contributed to the problem resolution by providing information, such as other aspects to consider, similar to the example mentioned by another developer here:

> "COMPANY identifies AT&T, Excite@Home but it also identifies R&D, AD&D, Q&A all are not really COMPANY. So there's a semantic error in the name of the rule (I know we shouldn't refer to the names too strictly, but still). ***[KNOWLEDGE CONTRIBUTION FOR PROBLEM RESOLUTION]***"

In this specific instance, the additional knowledge that was provided by another user completely changed the recommended solution to the removal of the original rule. This exemplifies how this integral innovation process, which utilized information technologies to exchange knowledge from various internal and external community members, helped create more effective and innovative solutions for software problems.

Integral open innovation process was used also for externalizing one's tacit knowledge about how open source software development worked in general, and how the given open source software development community operated, more specifically. The archival data provided ample evidence for how developers mentored others on both the software-based knowledge and community processes. An experienced developer mentioned:

> *"Now I spend most of my time mentoring others. Teaching them about how we do things around here, how to develop better software, etc."*

Peripheral Open Innovation Process

The Stable project used peripheral innovation process 16% of the time and the Delta project, 20% of the time. Peripheral open innovation process referred to the surface level knowledge creation in the form of combination and internalization.

Combination is defined as sorting, adding, categorizing existing information that is already made explicit. Internalization involves creation of documents, manuals or stories based on new knowledge whereby individuals convert explicit knowledge into tacit knowledge by means of self-reflection. The community members used the JIRA tool to communicate once more what was already observable to the community members through other information technologies. Thus, they combined different knowledge sources. This also enabled them to create systemic knowledge source, which was then used as a source to build other systems on for the purpose of obtaining more business intelligence on the project.

Writing documentation, such as documenting changes, writing tests, developing web page and wiki's were the ways in which the community members internalized the knowledge of others. When read by others, these documentations enabled third parties to internalize knowledge created by community members. Internalization happened through contributions to websites and wiki's, which were all submitted through SVN.

Leaders' Developers' and Users' Contributions to the Open Innovation Process

This section describes the findings on how community members who played different roles at the project contributed to open innovation process. Figure 2 shows the number of open innovation behaviors manifested during the regular and critical events.

At the Stable project, during the critical events, the leaders contributed 46%, 54% and 60% of the action-embedded, integral and peripheral open innovation processes respectively, whereas the developers contributed 44%, 44% and 40% of these processes and users made the negligible contributions of 10%, 2% and 0%.

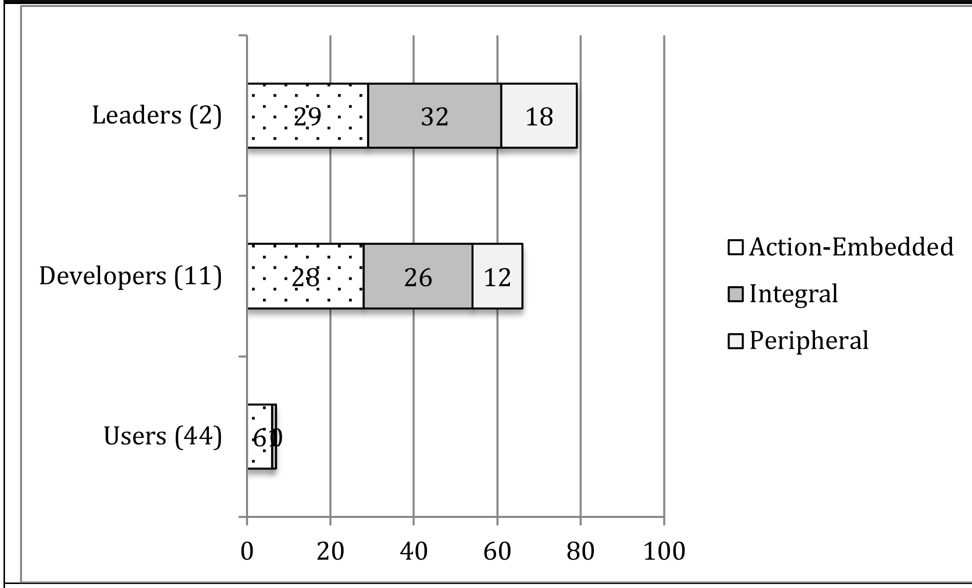
During the regular time period, the focus of the various community members seemed to have changed. The leaders contributed 5%, 45% and 28% of the action-embedded, integral and peripheral open innovation processes respectively. The developers, on the other hand contributed 60%, 47% and 72%. Finally, the users contributed 30%, 8% and 0%.

Figure 3 shows the number of open innovation behaviors that the Delta project members exhibit during the crucial and regular time periods. The Delta project showed similar patterns of leaders focusing very highly on all 3 types of open innovation behaviors during the critical time periods (They contributed 36%, 33% and 34% of these innovations respectively). Yet, Lucene leaders significantly reduced their action-embedded innovation behaviors during the regular time periods. They contributed only 5% of these behaviors. They focused instead on the integral innovation by contributing 25% of innovative behaviors.

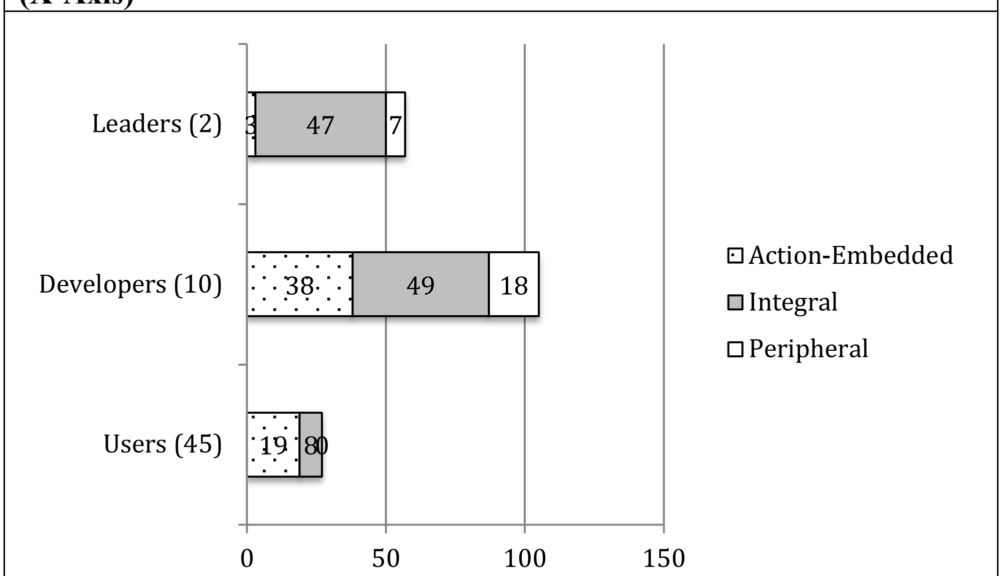
The key difference between the Stable and Delta projects during the first time period was the higher contributions of the Stable community leaders to all three types of innovation

Journal of Leadership and Management®
Cabell's Listed
An International Journal of Strategic, Behavioral and Public Aspects of Leadership in Management
A journal of the Institute of Leadership and Management Association, PO Box 564, Douglassville PA 19518. 484-332-3331. https://www.jleadershipmanagement.org

Figure 2. Contribution of Different Stable Members to Various Open Innovation Processes



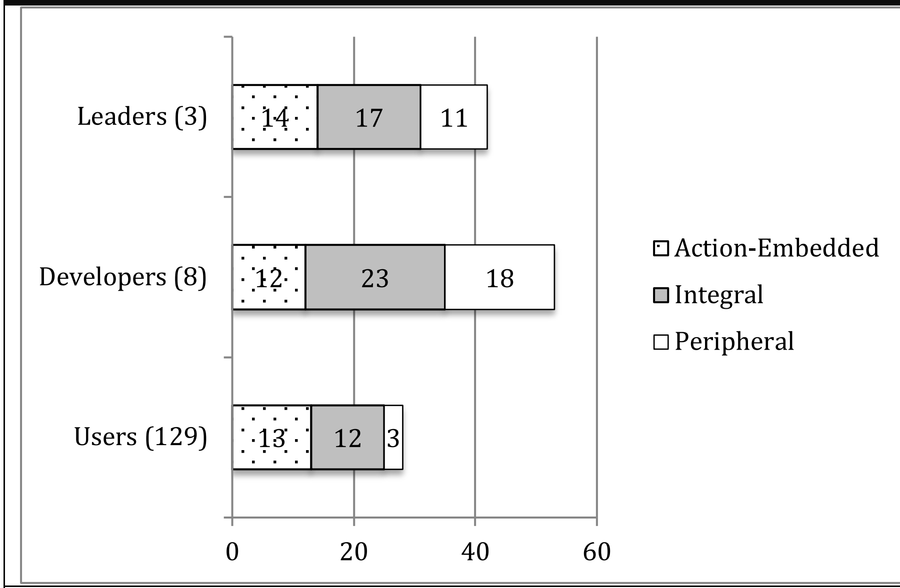
STABLE PROJECT CRUCIAL PERIOD
The Number of Open Innovation Behaviors During the Critical Events (X-Axis)



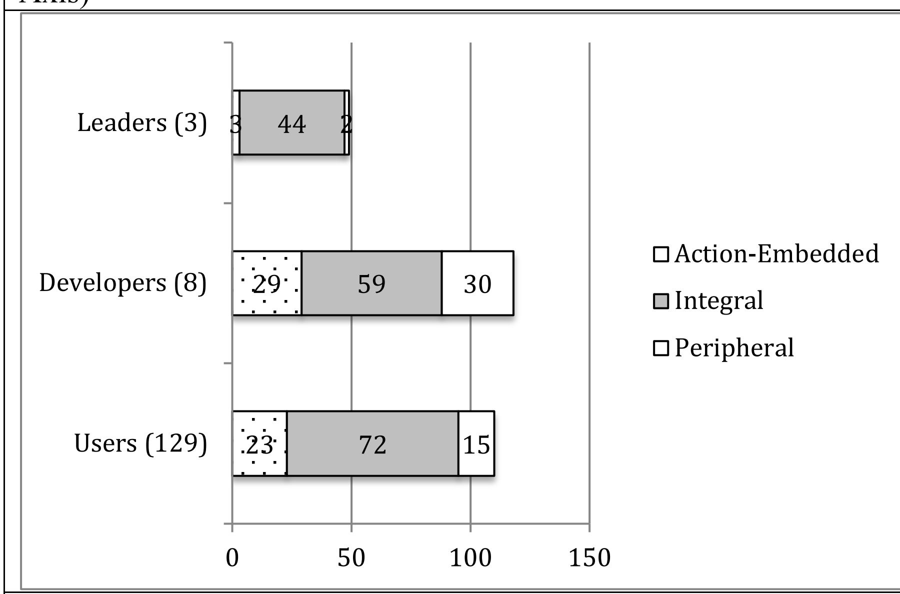
STABLE PROJECT REGULAR PERIOD
The Number of Open Innovation Behaviors During the Regular Events (X-Axis)

Journal of Leadership and Management®
An International Journal of Strategic, Behavioral and Public Aspects of Leadership in Management
Cabell's Listed
A journal of the Institute of Leadership and Management Association, PO Box 564, Douglassville PA 19518. 484-332-3331. https://www.jleadershipmanagement.org

Figure 3. Contribution of Different Delta Members to Various Open Innovation Processes



DELTA PROJECT CRUCIAL PERIOD
The Number of Open Innovation Behaviors During the Critical Events (X-Axis)



DELTA PROJECT REGULAR PERIOD
The Number of Open Innovation Behaviors During the Regular Events (X-Axis)

processes. Instead, the Delta community leaders contributed roughly about 30% of each of types of innovation processes. The second difference was the high level of involvement by the users in the innovation process.

Change in Open Innovation Communities Over Time

At the time of the case selection, a relatively major change was expected in the Delta project, since the founder and the existing leaders of the project were becoming more and more inactive over time. Further, new and active participants had joined the team. The Stable project had two highly hands-on leaders at the time of the case selection. While the project had attracted new members, only one of them was involved full-time at the project.

Changes in the Delta Community

Despite their lowered overall activity, the leaders of the Delta community (referred to as "tenured leaders" here forth) consistently provided about 30% of all three types of open innovation behaviors during both of the critical event periods. These tenured leaders also reduced their action-embedded innovation behaviors to about 5-6% across both regular event periods.

The Delta project kept a very innovative extended community, who exhibited about 30-40% of all three types of innovation behaviors during both of the regular time periods. The extended community contributions to open innovation were lower for the critical events. Especially for the peripheral open innovation, the contribution of the Delta users was as low as 9% of all contributions for the first time period and 5% in the second time period. However, for the other two categories (action-embedded and integral innovation respectively), the contribution of the Delta users were 33% and 23% for the first critical event period, and 20% and 43% for the second critical event period. Besides the tenured leaders, 5 Delta developers emerged as leaders as identified by the key informants in the second time period. Out of these five, 4 of them were actively innovating in the second regular time period, by exhibiting four or more open innovation behaviors. Very interestingly, some Delta users also contributed to innovation far above the typical user profile. Table 4 below shows the number of extended community members (users), who innovated beyond the typical user profile.

We refer to these users as innovative users. Innovative users exhibited four or more innovation behaviors, whereas the typical innovation behavior was around zero to one, and very rarely 3 or 4 per user. As can be seen from Table 5, these extended community members contributed to the community-wide innovation in ways that cannot be disregarded. The innovative users' contributions to innovation were higher during the regular periods for the action-embedded and peripheral innovation. On the other hand, the innovative users contributed more to integral innovation process during the critical periods.

Table 4. The Number of Highly Innovative Extended Community Members (Users) Over Time for Delta Project			
Number of Users Who Innovate Beyond the Regular User Profile			
Critical Events (Time Period 1)	2	Critical Events (Time Period 2)	4
Regular Events (Time Period 1)	12	Regular Events (Time Period 2)	9

If we distinguish between the emergent leaders who were more innovative versus those who innovated as much as a typical user did, we come up with a category of emergent leaders, which can be named innovative emergent leaders. During the critical events, this group of innovators contributed not nearly as much as the tenured leaders, but they came close. For example, they contributed 18 % of the action embedded innovation compared to 34% by developers; 12% of the integral innovation (compared to 31% by tenured leaders) and 25% of the peripheral innovation (as compared to 27% by the tenured leaders). Yet, during the regular event periods, the innovative emergent leaders contributed more to all innovation categories than tenured leaders.

Changes in the Stable Community

As expected, the leaders of the Stable community (referred to as "tenured leaders" here forth) provided high level of innovation in the first time period. During the critical events in the first time period, they contributed 46%, 54% and 60% of the whole community's action-embedded, integral and peripheral innovation behaviors respectively. Similar to the Delta team, the tenured leaders exhibited fewer innovation behaviors during the regular events of the first time period. For example, they reduced their action-embedded innovation behaviors to about 5% during the first regular event period.

The Stable project had roughly about 40 active users in their extended community. However, the contribution of this extended community to innovation was remarkably lower than that of the Delta project. (One may argue that this was normal since the Delta community had a community that was about 3 times larger. However, even when the contributions of the active Stable users were proportionally adjusted to account for the community size differences, they were much lower than those of the Delta extended community). For example, during

Journal of Leadership and Management®
An International Journal of Strategic, Behavioral and Public Aspects of Leadership in Management
Cabell's Listed
A journal of the Institute of Leadership and Management Association, PO Box 564, Douglassville PA 19518. 484-332-3331. https://www.jleadershipmanagement.org

both of the critical periods, the extended community's contribution to action-embedded innovation was 10% and to integral innovation was 2% and to peripheral innovation was non-existent. There were no users who innovated beyond the expected user profile during the critical events. During the regular events of two time periods, respectively 2 and 1 highly innovative users emerged (Table 6).

Table 5. The Contribution of Various Delta Project Participants to Open Innovation Process in the Second Time Period

% CONTRIBUTION TO OPEN INNOVATION PROCESS	Tenured Leaders	Innovative Emergent Leader/Developers	Non-Innovative Emergent Leader/Developers	Innovative Developers (never leaders)	Non-Innovative Developers (also never leaders)	Innovative Users (Users with >=4 innovative behaviors)	Non-innovative users	Everyone
Action-Embedded	34%	18%	0%	28%	0%	9%	11%	100%
Integral	31%	12%	1%	14%	0%	34%	9%	100%
Peripheral	27%	25%	0%	43%	0%	2%	2%	100%
Total Contribution	31%	17%	0%	25%	0%	19%	8%	100%
% of Population	2%	3%	1%	1%	1%	3%	89%	100%

DELTA PROJECT CRITICAL PERIOD #2
The Percentage of Open Innovation Behaviors During the Critical Events at Time=2

% CONTRIBUTION TO OPEN INNOVATION PROCESS	Tenured leaders	Innovative Emergent Leader/Developers	Non-Innovative Developers	Innovative Users (Users with >=4 innovative behaviors)	Non-innovative users
Action-Embedded	6%	54%	2%	27%	11%
Integral	29%	36%	1%	24%	10%

Peripheral	10%	63%	0%	5%	22%
Total Contribution	24%	41%	1%	23%	11%
% of Population	2%	3%	2%	6%	86%

DELTA PROJECT REGULAR PERIOD #2
The Percentage of Open Innovation Behaviors During the Regular Events at Time=2

Table 6. The Number of Highly Innovative Extended Community Members (Users) Over Time for Stable Project

	Number of Users Who Innovate Beyond the Regular User Profile		
Critical Events (Time Period 1)	0	Critical Events (Time Period 2)	0
Regular Events (Time Period 1)	2	Regular Events (Time Period 2)	1

The Stable team showed a remarkable difference in its trend during the second time period (Table 7), compared to the trends observed in the first time period as well as compared to the other project. First of all, the tenured leaders reduced their innovation behaviors remarkably during the critical events of the second time period (12%, 17% and 18% respectively for action-embedded, integral and peripheral innovation), yet they kept innovating more than their usual patterns discussed earlier (31%, 55% and 28% respectively for action-embedded, integral and peripheral innovation).

Journal of Leadership and Management®
An International Journal of Strategic, Behavioral and Public Aspects of Leadership in Management
Cabell's Listed
A journal of the Institute of Leadership and Management Association, PO Box 564, Douglassville PA 19518. 484-332-3331. https://www.jleadershipmanagement.org

Table 7. The Contribution of Various Stable Project Participants to Open Innovation Process in the Second Time Period

% CONTRIBUTION TO OPEN INNOVATION PROCESS	Leaders (Old)	Innovative developers	Non-innovative developers	Innovative users	Non-innovative users
Action-Embedded	12%	76%	2%	0%	10%
Integral	17%	77%	5%	0%	2%
Peripheral	18%	82%	0%	0%	0%
Total Contribution to Innovation	16%	78%	3%	0%	4%
Number of People	2	4	9	0	41

STABLE PROJECT CRITICAL PERIOD #2

The Percentage of Open Innovation Behaviors During the Critical Events at Time=2

% CONTRIBUTION TO OPEN INNOVATION PROCESS	Tenured Leaders	Innovative developers	Non-innovative developers	Innovative user	Non-innovative users
Action-Embedded	31%	37%	4%	16%	12%
Integral	55%	24%	3%	3%	14%
Peripheral	28%	44%	11%	11%	6%
Total	46%	29%	4%	7%	13%
Count (# of Individuals Involved)	2	4	9	1	40

STABLE PROJECT REGULAR PERIOD #2

The Percentage of Open Innovation Behaviors During the Regular Events at Time=2

However, the innovation gap that is left by the tenured leaders was filled by innovative developers in the second time period. Interestingly an innovative user and four innovative developers contributed quite a bit to the regular events in the second time period, which changed the nature of innovation distribution within the team.

Another notable change in the Stable team is the leader-like behavior of one of the developers. This developer contributed 61%, 40% and 62% to the three innovation processes during the second time period's critical events and 6%, 12% and 6% to the innovation processes during regular events. In that sense, this developer's innovation patterns followed that of a leader. Yet the developer was not identified as an emergent leader.

Table 8. Innovation Per Thread for Both Communities Over Time

	Critical Events (Time Period 1)	Regular Events (Time Period 1)	Critical Events (Time Period 1)	Regular Events (Time Period 2)
Stable Community	2.3	2.4	1.9	2.4
Delta Community	3.2	3.5	4.5	5.4

In an effort to understand the influence of various changes to both communities, the level of innovations per thread of data is analyzed (Table 8). An increase in average innovation per period was observed in the Delta Community. The Stable community showed about the same level of innovation with the exception of the critical events of the second time period, where one developer innovated much more than usual and where the leaders visibly reduced their innovations.

Discussion

This paper contributed to opening the black box of open innovation process. Specifically, this study addressed how open innovation community leaders, members and extended community contribute to open innovation through information technologies as the community dynamics change. The following sections present theoretical and practical contributions of these findings.

Contributions to Theory and Future Research

Organizations increasingly aim at emulating the open source software development community innovation model. Yet, emulation requires the understanding of what constitutes innovation, and how leaders and community members can provide such innovation (Agerfalk

& Fitzgerald, 2008). For the community members, the goal is not only to innovate, but also to elicit innovation behaviors from other local and extended community members.

Previous literature on open source software development communities identified the following roles for different community members: The leaders strategically influenced software development through action-embedded transformational leadership (Eseryel & Eseryel, 2013). By directly contributing to software development in the form of work, they increased software development effectiveness and transformed technology vision (Eseryel & Eseryel, 2013). The users, on the other hand, contributed specifically to software quality improvement by generating patches and reporting bugs (Setia, et al., 2012), which is also part of socialization type of knowledge creation (Eseryel, in press) according to Nonaka and Takeuchi's (1995) book on innovation. While this current study confirmed these earlier findings, it also provided additional insights in the form of a finer grained understanding of how these different community members contributed to innovation, and how their contributions changed over time as the community continued to evolve.

First, a general trend of the leaders was to focus their innovation activities on the critical events. This may likely be due to the leaders' high level of expertise, and their ability and interest in resolving these issues. The leaders showed a general trend of a visible decrease in innovation in general and action-embedded innovation in particular during the regular events. Previous literature had created an expectation of high action-embedded innovation by leaders (Eseryel, 2013), not accounting for the differences in the types of software development activity (critical versus regular). This study determined integral innovation to be the most frequently used type of innovation adopted by the leaders. While action-embedded innovation followed in the second rank, the leaders' action-embedded innovation focused more on crucial events, leaving room for developers and users to take on more initiatives in this area.

The gap in innovation, caused by lessened activity of leaders during the regular events is often filled by the developers and extended community members. It may be that the leaders purposefully reduce their innovations in regular events to encourage the inputs of the rest of the community members. Alternatively, the high level of existing community innovation regarding the regular events may be enabling the leaders to focus their innovation efforts to more critical events.

Next, the contributions of the extended community members (i.e. the users) are discussed. Eseryel (2014) had suggested that the little innovation contributed by each of the extended community members mattered. She observed that especially for the communities with large user bases, small contributions may add up to 50% the knowledge creation (Eseryel, in press). However, the two communities in this study differed in their contributions to innovation, even after adjustments were made to their level of innovation to account for difference in community sizes. The open innovation community (Delta) whose leaders provided about 30% of all innovation types consistently through critical issues also presented high level of user innovation. During the regular events when the Delta leaders did not contribute much to innovation, the users exhibited high levels of innovation.

The Delta community was originally identified for this study with the expectation of a leader and member change. As expected, the tenured leaders of the community reduced their activities in general, and five new leaders emerged over time. Yet, very surprisingly, the tenured leaders' innovativeness stayed constant despite lowered activities. With the addition of the new leaders who were innovating at a higher rate than before, the community's overall innovation per thread increased about 40% for the critical event threads and 54% for the regular event threads. Thus, the constant contribution of the existing leaders to innovation, and enabling the existing community members (developers) to innovate at an higher level, helped increase the community innovativeness. While it was expected that the leadership change may influence the innovation negatively, at least in the short-term, the opposite was the case. Thus, the study showed that the increased rate of innovation by developers together with constant level of innovation by the leaders help improve the community innovativeness. Future research should investigate how leaders can further energize developer innovation through their own behaviors.

Similarly, this study indicated an extension of the extant research on the importance of users in open innovation communities: While the users' small innovations add up to make a difference (Eseryel, 2014), also single users can take initiatives to contribute very significantly to the innovativeness of the community. This was exemplified during the regular events of the second time period within the Stable community, where one user provided the 29%, 17% and 17% of the community's innovations across three types of innovations. To indicate the significance of this user's innovation, it can be highlighted that his/her contribution is similar to the level of innovation by the two leaders during the crucial events of the same time period. Thus, while enlarging user base is crucial, it is also crucial to tap into this extended community to motivate those individuals who have the knowledge to innovate to contribute in more significant ways to innovation. Future research should investigate how leaders can motivate extended community members to innovate in a significant way. Thus, we would like to add to the previous recommendation of researchers to engage and harvest external knowledge of external community members (von Krogh et al., 2003; West & O'Mahony, 2005). In an effort to engage and harvest external community member knowledge, both enlargement of the community and enablement of strong innovation by the developers and users in the community are important.

It was observed that the members of the internal community (developers) may replicate the innovativeness level of their leaders. This was observed in both communities, where in one community, the innovative members emerged as leaders and in the other, this did not happen. In one community, also one non-innovative developer emerged as a leader. This raises the following important theoretical and practical questions: On the theoretical level, the question becomes "What is the connection between innovativeness and leader emergence in open innovation communities? On the practical level, the key question for community leaders is "How can the information technologies and existing leaders support the increased emergence of innovative leaders?"

A very interesting finding of this study is the fact that the more dynamic team (Delta) exhibited more stable and improving innovativeness than the team, which was more stable in its membership and leadership. This finding has two implications; (1) First, the activity level and innovativeness level may differ. Thus, the theory should look at the connections between them. (Even though the activity level of the leaders changed in the Delta team, the leaders kept their innovation patterns similar over time, thus making consistent innovation support by leaders important for overall community innovation.) (2) Secondly, perhaps this indicates or reminds us that open innovation communities, by definition, go through constant change over time. Perhaps we can say that there are no open innovation teams stable in their membership and leadership. This implication is crucial for researchers in that the methods used to investigate open innovation should take into account a constant change in community innovation activities. This knowledge would favor longitudinal or cross-sectional analysis over studies at a single point in time for the generalizability of the findings.

How Information Technology Changes the Type of Knowledge Created Through Various Innovation Types. This paper contributes to open innovation literature by determining unique types of innovation (action-embedded, integral and peripheral innovation) that are applicable to open innovation communities, which operate nearly extensively through information technologies. This study helps future researchers operationalize these innovation behaviors by providing specific ways in which these three innovation behaviors can be observed in the archival information technology artifacts (mailing lists, issue tracker and software contributions) that are created by the community members.

Despite the abundance of literature that built on Nonaka and Takeuchi's (1995) work, the actual empirical operationalization of the knowledge creation modes was rather rare (e.g., Nonaka, Byosiere, Borucki, & Konno, 1994).

Similarly, the identification of these knowledge creation modes in IT-enabled open innovation settings was just as rare (e.g., Eseryel, 2014). Thus, this article built on previous knowledge created by Nonaka and Takeuchi (1996) and Eseryel (2014) to capture innovation processes within IT-enabled open innovation communities. Table 9 presents how the open innovation over the information technologies manifested here differently than that discussed by Nonaka and Takeuchi (1995). The differences between the non-IT mediated communication and collaboration contexts of the brick-and-mortar companies (used by Nonaka and Takeuchi) and the IT-enabled open innovation are described below.

Table 9. Comparison of the Knowledge Types Created in Open Innovation Contexts to Knowledge Types Created in Non-IT Mediated Settings

Innovation Type	**Knowledge-Creation Mode** (Nonaka & Takeuchi, 1995)	**Type of Knowledge Created in Non-IT Mediated Contexts** (Nonaka & Takeuchi, 1995)	**Type of Knowledge Created in IT-Enabled Open Innovation**
Action-embedded	Socialization	Sympathized Knowledge	Sympathized Knowledge & Operational Knowledge
Integral	Externalization	Conceptual Knowledge	Interactive Integral Knowledge
Peripheral	Combination	Systemic Knowledge	Systemic Knowledge & Operational Knowledge
Peripheral	Internalization	Operational Knowledge	Operational Knowledge

In IT-enabled open-innovation setting, the action-embedded innovation happened through interaction with the information technology artifacts, rather than through interaction and observation. The observation in physical settings created sympathized knowledge, which produced shared mental models (Nonaka & Takeuchi, 1995). In IT-enabled open innovation, both sympathized knowledge and operational knowledge was created. The community members established shared mental models on software development (Scozzi, Crowston, Eseryel, & Li, 2008). And they improved their understanding of the software artifact, which was a key part of their operations.

The externalization knowledge-creation mode in non-IT mediated contexts constituted purely one-way transfer of conceptual knowledge in the form of metaphors and analogies. Metaphors and analogies were also observed in some open source software development communities such as KDE (Hemetsberger & Reinhardt, 2006). However, Nonaka et al.

(1994) could not replicate this finding. Furthermore, metaphors and analogies were not found in the Lucene project (Eseryel, in press) as well as in the Apache projects used for this current investigation. Innovation through externalization was defined as integral innovation in this study, because the innovation that was observed in this study involved a constant multi-way interaction rather than one-way dissemination of conceptual knowledge. The knowledge that was created enabled by the threading features of the information technologies such as mailing lists and issue trackers was different: Namely these features enabled multi-way interaction, and creation of knowledge that built on other community members' knowledge. Thus, the conceptual knowledge became operationalized and built on. Thus, we named the type of knowledge that is created through the integral innovation process "interactive integral knowledge" to reflect that it was developed through a multi-way community process (thus 'interactive), and that it integrated knowledge of multiple community members (thus 'integral').

Two modes of knowledge creation together made up the peripheral innovation. This study showed that the knowledge type that was developed as a result of internalization was the same in open innovation communities with that in non-IT mediated communities. This happened because the main behavior for internalization was the same: In IT-enabled open innovation communities, the key internalization behavior was also documentation. Yet, perhaps differently than the non-IT mediated contexts, the documentation was created through information technologies, and disseminated through information technologies. This enabled with wider dissemination of the knowledge, and thus enabled "open innovation" by transferring internal knowledge to the extended community in an easy, fast and cheap way. Yet, the information technologies did not influence the type of knowledge created.

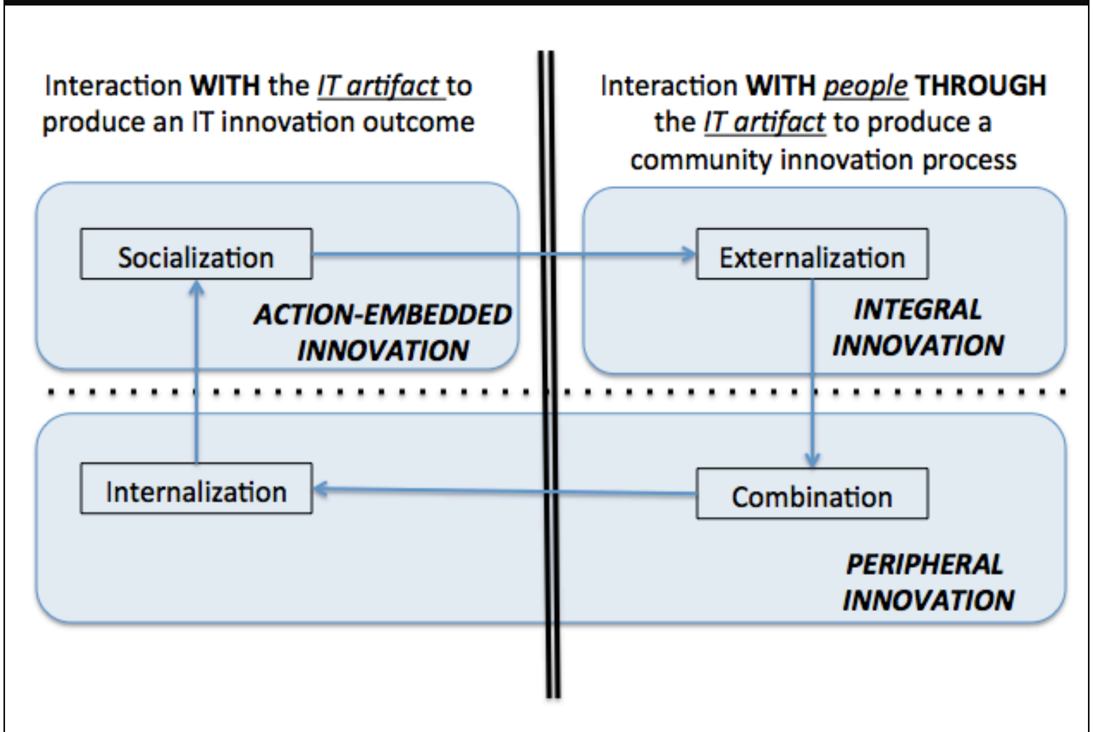
However, the peripheral innovation that was done through combination knowledge creation mode showed some differences in the type of knowledge created. The combination knowledge creation mode typically referred to combination of different knowledge sources. This created opportunities to build systems on the combined knowledge, thus creating systemic knowledge through information technologies. In the open innovation communities, the community members chose to bring back different types of knowledge into information technologies such as issue trackers. Further, members turned on software features that enabled other members to instantly be aware of changes in issue trackers or work contributions by creating central mailing lists that automatically reported changes. These features disseminated knowledge to the extended community, thereby also increasing their operational knowledge.

To sum up, the use of information technology in open innovation communities increased the creation of operational knowledge in two of the 4 types of knowledge creation modes and changed the type of knowledge created in one type of knowledge creation mode (Table 9).

Lastly, the way in which community members used the IT artifact changed depending on the type of knowledge creation (Table 10). During socialization and internalization, individuals interacted *with* the IT artifact in order to produce IT innovation. Their innovation process resulted in the creation of an IT artifact, such as a software patch or a digital documentation.

On the other hand, during externalization and combination, community members interacted with each other *through* the IT artifact. This enabled them to create a community innovation outcome, such as problem solving, or decision-making, or transfer of operational knowledge, which is used for software development.

Table 10. Comparison of the Knowledge Types Created in Open Innovation Contexts to Knowledge Types Created in Non-IT Mediated Settings



Contributions to Practice

This paper contributes to the practice by explicating how open innovation communities innovate through information and communication technologies. The implication of this study for information systems managers and practitioners, who incorporate open source software development teams into their organizational practice, is the need for: (1) supporting the existing users and encouraging them to stay involved, in an effort to keep the number of external knowledge contributors high; (2) keeping a small number of highly skilled developers who can deal with critical events.; and (3) getting the extended community members more active in the regular time periods, especially with regard to action-embedded innovation. This may inform which skills these organizations focus on during recruitment and training of team members (such as action focus, skills on software development, etc.).

The managers/leaders should be aware that by investing in developers and uses, it is possible to increase the group innovativeness. However, increasing group innovativeness would require that the leaders find a way to keep them innovative, while also enabling others to innovate. This is certainly a difficult balancing act.

Lastly, this research showed that part of the innovation activities involve interaction with the IT artifact to produce and IT innovation outcome. On the other hand, other innovation includes using the IT artifact as a tool to interact with people to produce a community innovation process. Therefore, the development of strong IT skills is key to open innovation communities. Furthermore, having a variety of information technologies with features that enable all three types of innovation is the required basis for a healthy open innovation community.

Limitations

This paper presented findings from comparative case studies. A limitation of this study is the fact that it presents a small number of case studies. The findings of this study can thus be generalized to similar contexts, where the task is highly technical and knowledge-intensive, where the team is distributed globally and highly dependent on information technologies. The findings of this study can also be generalized to the companies that are using open source practices inside the environment of a company, i.e., inner sources. For example, Morgan et al. (2011) analyzed a medical equipment and device supplier, which utilizes open source practices in the distributed development of their product line. This presents an example of a company, which has a highly technical and knowledge-intensive task of medical equipment development, a team that is distributed between the U.S. and the Netherlands, and high dependence on information technologies.

REFERENCES

Agerfalk, P. B., & Fitzgerald, B. (2008). Outsourcing to an unknown work- force: Exploring opensourcing as a global sourcing strategy. *MIS Quarterly, 32*(2), 385-410.

Ali, A., Krapfel, R., & LaBahn, D. (1995). Product innovativeness and entry strategy: Impact on cycle time and break-even time. *Journal of Product Innovation Management, 12*, 54-69.

AlMarzouq, M., Zheng, L., Rong, G., & Grover, V. (2005). Open source: Concepts, benefits, and challenges. *Communications of the Association for Information Systems, 16*(37), 756-784.

Atuahene-Gima, K. (1995). An exploratory analysis of the impact of market orientation on new product performance: A contingency approach. *Journal of Product Innovation Management, 12*, 275-293.

Banker, R. D. (2007, April 09). [Email exchange with the author on research method used for the article "The Evolution of Research on Information Systems: A Fiftieth-Year Survey of the Literature in Management Science."].

Bonaccorsi, A., & Rossi, C. (2006). Comparing motivations of individual programmers and firms to take part in the open source movement: from community to business. *Knowledge, Technology & Policy, 18*(4), 40-64.

Brooks, F. P. (1995). No silver bullet: essence and accidents of software engineering. In *Mythical Man-Month*.

Butler, B. (2001). Membership size, communication activity, and sustainability: A resource-based model of online social structures. *Information Systems Research, 12*(4), 346-362.

Cebon, P., & Newton, P. (1999). *Innovation in Firms: Towards a Framework for Indicator Development*. Melbourne Business School.

Chandy, R. K., & Tellis, G. J. (2000). The incumbents curse: incumbency, size, and radical product innovation. *Journal of Marketing, 64*, 1-17.

Chesbrough, H. M. (2003). *Open Innovation*. Boston, MA: Harvard Business School Press.

Colarelli O'Connor, G. (1998). Market learning and radical innovation: A cross case comparison of eight radical innovation projects. *Journal of Product Innovation Management, 15*(2), 151-166.

Comino, S., & Manenti, F. M. (2005). Government policies supporting open source software for the mass market. *Review of Industrial Organization, 26*(2), 217-240.

Conboy, K., & Morgan, L. (2011). Beyond the customer: Opening the agile systems development process. *Information and Software Technology, 53*(5), 535-542. doi:http://dx.doi.org/10.1016/j.infsof.2010.10.007

Cook, I., & Horobin, G. (2006). Implementing eGovernment without promoting dependence: open source software in developing countries in Southeast Asia. *Public Administration and Development, 26*(4), 279-289. doi:10.1002/pad.403

Cooper, R. G. (1979). The dimensions of industrial new product success and failure. *Journal of Marketing, 43*, 93-103.

Cooper, R. G., & de Brentani, U. (1991). New industrial financial services: What distinguishes the winners. *Journal of Product Innovation Management, 8*(75-90).

Corbin, J., & Strauss, A. (2008). *Basics of Qualitative Research: Techniques and Procedures for Developing Grounded Theory* (3rd ed.). Thousand Oaks, CA: Sage.

Cougar, J. (1990). Ensuring creative approaches in information system design. *Managerial and Decision Economics, 11*(2), 281-295.

Cox, A. (1998). Cathedrals, bazaars and the town council. Retrieved from http://slashdot.org/features/98/10/13/l423253.shtml

Crossan, M. M., & Apaydin, M. (2010). A multi-dimensional framework of organizational innovation: A systematic review of the literature. *Journal of Management Studies, 47*(6), 1154-1191.

Crowston, K., & Annabi, H. (2005). *Effective work practices for FLOSS development: A model and propositions.* Paper presented at the In Thirty-Eighth Hawaii International Conference on System Science (HICSS-38). Kona, HI.

Crowston, K., & Howison, J. (2005). The social structure of free and open source software development. *First Monday, 10*(2). Retrieved from http://firstmonday.org/htbin/cgiwrap/bin/ojs/index.php/fm/ article/viewArticle/1207/1127

Crowston, K., & Howison, J. (2006). Hierarchy and centralization in free and open source software team communications. *Knowledge, Technology & Policy, 18*(4), 65-85. Retrieved from http://search.ebscohost.com/login.aspx?direct=true&db=tfh&AN=21500199&site=ehost-live

Crowston, K., Howison, J., Masango, C., & Eseryel, U. Y. (2005, 7-10 August). *Face-to-face interactions in self-organizing distributed teams.* Paper presented at the OCIS Division, Academy of Management Conference, Honolulu, HI.

Davenport, T. H. (1997). Knowledge management at Ernst & Young. *Knowledge Management Case Study.* Retrieved from http://www.bus.utexas.edu/kman/e_y.htm

Dinkelacker, J., Garg, P. K., Miller, R., & Nelson, D. (2002). *Progressive open source.* Paper presented at the Proceedings of the International Conference on Software Engineering (ICSE '02), Orlando.

Ducheneaut, N. (2005). Socialization in an open source software community: A socio-technical analysis. *Computer Supported Cooperative Work, 14*(4), 323-368.

Durmusoglu, S. S., & Barczak, G. (2011). The use of information technology tools in new product development phases: Analysis of effects on new product innovativeness, quality, and market performance. *Industrial Marketing Management, 40*(2).

Eisenhardt, K. M. (1989). Building Theories from Case Study Research. *Academy of Management Review, 14*(4), 532-550.

Eseryel, U. Y. (2013). *Information Technology Self-Leadership*. Department of Innovation Management & Strategy. University of Groningen. Unpublished Work.

Eseryel, U. Y. (2014). IT-Enabled Knowledge Creation for Open Innovation. *Journal of the Association for Information Systems 15*(11), 805-834. doi:10.17705/1jais.00378

Eseryel, U. Y., & Eseryel, D. (2013). Action-embedded transformational leadership in self-managing global information technology teams. *The Journal of Strategic Information Systems, 22*(2), 103-120.

Fielding, R. T. (1999). Shared leadership in the Apache project. *Communications of the ACM, 42*(4), 42-43.

Gacek, C., Lawrie, T., & Arief, B. (2001). *The many meanings of open source* (1). Retrieved from http://www.dirc.org.uk/ publications/techreports/papers/1.pdf

Garcia, R., & Calantone, R. (2002). A critical look at technological innovation typology and innovativeness terminology: a literature review. *Journal of Product Innovation Management, 19*(2), 110-132. doi:10.1111/1540-5885.1920110

Gassmann, O., & Enkel, E. (2001). Towards a theory of open innovation: three core process archetypes. Retrieved from http://www.alexandria.unisg.ch/Publikationen/274

Goldenberg, J., Lehmann, D. R., & Mazursky, D. (1999). *The primacy of idea itself as a predictor of new product success*. Marketing Science Institute Working Paper.

Goldman, R., & Gabriel, R. P. (2005). *Innovation Happens Elsewhere: Open Source as Business Strategy*. San Francisco: Morgan Kaufmann.

Green, S. G., Gavin, M. B., & Aiman-Smith, L. (1995). Assessing a multidimensional measure of radical technological innovation. *IEEE Transactions on Engineering Management, 42*(3), 204-213.

Guba, E. G., & Lincoln, Y. S. (1994). Competing paradigms in qualitative research. In N. K. Denzin & Y. S. Lincoln (Eds.), *Handbook of Qualitative Research* (pp. 105-117). Thousand Oaks: Sage.

Haefliger, S., von Krogh, G., & Spaeth, S. (2008). Code Reuse in Open Source Software. *Management Science, 54*(1), 180-193. Retrieved from http://libraries.ou.edu/access.aspx?url=http://search.ebscohost.com/login.aspx?direct=true&db=buh&AN=29984790&site=ehost-live

Hare, A. P. (1976). *Handbook of small group research* (2nd ed.). New York: Free Press.

Hars, A., & Ou, S. (2001). *Working for Free? - Motivations of Participating in Open Source Projects*. Paper presented at the Proceedings of the 34th Hawaii International Conference on System Sciences.

Heckman, R., & Misiolek, N. I. (2005). *Leaders and followers in student online project teams*. Paper presented at the 38th Hawaii International Conference on System Sciences. http://csdl2.computer.org/comp/proceedings/hicss/2005/2268/01/22680004c.pdf

Hedgebeth, D. (2007). Gaining competitive advantage in a knowledge-based economy through the utilization of open source software. *VINE, 37*(3), 284-294.

Helfat, C. E., & Peteraf, M. A. (2003). The dynamic resource-based view: Capability lifecycles. *Strategic Management Journal, 24*(10), 997-1010.

Hemetsberger, A., & Reinhardt, C. (2006). Learning and Knowledge-building in Open-source Communities: A Social-experiential Approach. *Management Learning, 37*(2), 187-214. doi:10.1177/1350507606063442

Hermann, S., Hertel, G., & Niedner, S. (2000). Linux Study Homepage. Retrieved from www.psychologie.uni-kiel.de/linux-study/writeup.html

Himanen, P., Torvalds, L., & Castells, M. (2001). *The Hacker Ethic*. New York: Random House.

Jansen, D. S., & Saiedian, H. (2006). *Test-driven learning: intrinsic integration of testing into the CS/SE curriculum.* Paper presented at the Proceedings of the 37th SIGCSE technical symposium on Computer science education, Houston, Texas, USA.

Jha, S. R. (2002). *Reconsidering Michael Polanyi's Philosophy*. Pittsburgh: University of Pittsburgh Press.

Kautz, K., & Thaysen, K. (2001). Knowledge, learning and IT support in a small software company. *Journal of Knowledge Management, 5*(4), 349-357.

Kleis, L., Chwelos, P., Ramirez, R. V., & Cockburn, I. (2012). Information technology and intangible output: The impact of IT investment on innovation productivity. *Information Systems Research, 23*(1), 42-59.

Kogut, B. (2000). The network as knowledge: Generative rules and the emergence of structure. *Strategic Management Journal, 21*(405-425).

Kohanski, D. (1998). *Moths in the Machine*. New York: St. Martin's Griffin.

Lapointe, L., & Rivard, S. (2005). A multilevel model of resistance to information technology implementation. *MIS Quarterly, 29*(3), 461-491.

Lazar, J., & Preece, J. (2002). Online communities: Usability, sociability and users' requirements. In H. van Oostendorp (Ed.), *Cognition in the Digital World* (pp. 127-151). Mahwah, NJ: Lawrence Erlbaum Associates, Inc.

Lee, G. K., & Cole, R. E. (2003). From a Firm-Based to a Community-Based Model of Knowledge Creation: The Case of the Linux Kernel Development. *Organization Science, 14*(6), 633-649.

Lee, M., & Na, D. (1994). Determinants of technical success in product development when innovative radicalness is considered. *Journal of Product Innovation Management, 11*, 62-68.

Lerner, J., & Triole, J. (2000). *The simple economics of open source*. NBER. Working Paper.

Levina, N., & Vaast, E. (2005). The emergence of boundary spanning competence in practice: implications for implementation and use of information systems. *MIS Quarterly, 29*(2), 335-363.

Ljungberg, J. (2000). Open source movements as a model for organizing. *European Journal of Information Systems, 9*(4), 208-216.

Losh, C., Stein, L., & Terwiesch, C. (1996). Measuring development performance in the electronics industry. *Journal of Product Innovation Management, 13*, 3-20.

Lundell, B., Lings, B., & Syberfeldt, A. (2011). Practitioner perceptions of open source software in the embedded systems area. *Journal of Systems and Software, 84*(9), 1540-1549. doi:10.1016/j.jss.2011.03.020

Ma, M., & Agarwal, R. (2007). Through a flass darkly: Information technology design, identity verification and knowledge contribution in online communities. *Information Systems Research, 18*(1), 42-67.

Miles, M. B., & Huberman, A. M. (1994). *Qualitative Data Analysis* (2 ed.). Thousand Oaks, CA: Sage.

Mockus, A., Fielding, R. T., & Herbsleb, J. D. (2002). Two case studies of Open Source Software development: Apache and Mozilla. *ACM Transactions on Software Engineering and Methodology, 11*(309-346).

More, R. A. (1982). Risk factors in accepted and rejected new industrial products. *Industrial Marketing Management, 11*, 9-15.

Morgan, L., Feller, J., & Finnegan, P. (2011). *Exploring inner source as a form of intraorganisational open innovation.* Paper presented at the ECIS 2011 Helsinki, Finland.

Morner, M., & von Krogh, G. (2009). A Note on Knowledge Creation in Open-Source Software Projects: What Can We Learn from Luhmann's Theory of Social Systems? *Systemic Practice and Action Research, 22*(6), 431-443.

Morse, J. M. (2004). Theoretical saturation. In M. S. Lewis-Beck, A. Bryman, & T. F. Liao (Eds.), *Encyclopedia of Social Science Research Methods* (Vol. 1-3). Thousand Oaks, CA: Sage Publications.

Nambisan, S., & Sawhney, M. (2007). *The Global Brain: Your Roadmap for Innovating Faster and Smarter in a Networked World*. Philadelphia: Wharton School Publishing.

Neuendorf, K. A. (2002). *The content analysis guidebook*. Thousand Oaks, CA: Sage Publications.

Nonaka, I., Byosiere, P., Borucki, C. C., & Konno, N. (1994). Organizational knowledge creation theory: A first comprehensive test. *International Business Review, 3*(4), 337-351.

Nonaka, I., & Takeuchi, H. (1995). *The Knowledge-Creating Company: How Japanese Companies Create the Dynamics of Innovation*. New York: Oxford University Press, inc.

Nonaka, I., & von Krogh, G. (2009). Tacit Knowledge and Knowledge Conversion: Controversy and Advancement in Organizational Knowledge Creation Theory. *Organization Science, 20*(3), 635-652. Retrieved from http://libraries.ou.edu/access.aspx?url=http://search.ebscohost.com/login.aspx?direct=true&db=buh&AN=42421326&site=ehost-live

Pavlou, P., & Sawy, O. (2006). From IT leveraging competence to competitive advantage in turbulent environments: The case of new product development. *Information Systems Research, 17*(3), 198-220.

Pawlowski, S. D., & Robey, D. (2004). Bridging User Organizations: Knowledge Brokering and the Work of Information Technology Professionals. *MIS Quarterly, 28*(4), 645-672.

Pettigrew, A. M. (1990). Longitudinal field research on change: theory and practice. *Organization Science, 1*(3), 267-292.

Potter, W. J., & Levine-Donnerstein, D. (1999). Rethinking validity and reliability in content analysis. *Journa of Applied Communication Research, 27*, 258-284.

Ransbotham, S., & Kane, G. C. (2011). Membership turnover and collaboration success in online communities: Explaining rises and falls from grace in wikipedia. *MIS Quarterly, 35*(3), 613-627.

Robillard, P. (1999). The role of knowledge in software development. *Communications of the ACM, 42*(10), 87-92.

Sarker, S., Grewal, R., & Sarker, S. (2002). *Emergence of leaders in virtual teams: what matters?* Paper presented at the Proceedings of the 35th Hawaii International Conference on System Sciences

Scozzi, B., Crowston, K., Eseryel, U. Y., & Li, Q. (2008, Jan 7-10). *Shared mental models among open source software developers.* Paper presented at the 41st Annual Hawaii International Conference on System Sciences (HICSS 2008), Hawaii.

Setia, P., Rajogopalan, B., & Sambamurthy, V. (2012). How peripheral developers contribute to open-source software development. *Information Systems Research, 23*(1), 144-163.

Souder, W. E., & Song, M. X. (1997). Contingent product design, and marketing strategies influencing new product success, and failure in US, and Japanese electronic firms. *Journal of Product Innovation Management, 14*, 21-34.

The Apache Software Foundation. (2012). The Apache Software Foundation Board of Directors. Retrieved from http://www.apache.org/foundation/board/

Thomas, D., & Hunt, A. (2004). Open source ecosystems. *IEEE Software, 32*(1), 89-91.

Trott, P., & Hartmann, D. (2009). Why 'open innovation' is old wine in new bottles. *International Journal of Innovation Management, 13*(4), 715-736.

von Hippel, E., & von Krogh, G. (2003). Open source software and the 'private-collective' model: Issues for organization science. *Organization Science, 14*(2), 209-223.

von Krogh, G. (2002). The communial resource and information systems. *Journal of Strategic Information Systems, 11*(2), 85-107.

von Krogh, G., & Spaeth, S. (2007). The open source software phenomenon: characteristics that promote research. *Journal of Strategic Information Systems, 16*(3), 236-253. doi:10.1016/j.jsis.2007.06.001

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

Journal of Leadership and Management®
An International Journal of Strategic, Behavioral and Public Aspects of Leadership in Management
Cabell's Listed
A journal of the Institute of Leadership and Management Association, PO Box 564, Douglassville PA 19518. 484-332-3331. https://www.jleadershipmanagement.org

Wasserman, A. I. (2009). Building a business of open source software. In C. Petti (Ed.), *Cases in Technological Entrepreneurship* (pp. 107-121). Salento, Italy: Edward Elgar Publishing.

Weber, S. (2004). *The Success of Open Source*: Harvard University Press.

West, J., & O'Mahony, S. (2005, January 3-6). *Contrasting community building in sponsored and community founded open source projects.* Paper presented at the 38th Annual Hawai'i International Conference on System Sciences, Waikoloa, Hawaii.