AN ABSORPTIVE CAPACITY PERSPECTIVE ON OPEN SOURCE SOFTWARE DEVELOPMENT GROUP PERFORMANCE

Alternative Approaches to Information Systems Development

Sherae L. Daniel
University of Maryland
College Park, MD U.S.A.
sdaniel@rhsmith.umd.edu

Ritu Agarwal
University of Maryland
College Park, MD U.S.A.
ragarwal@rhsmith.umd.edu

Katherine J. Stewart
University of Maryland
kstewart@rhsmith.umd.edu

Abstract

An organization’s absorptive capacity, or its ability to acquire and use knowledge, is important for facilitating innovation that can lead to improvements in organizational performance. We examine the effects of absorptive capacity on open source software development group performance. Specifically, in addition to the commonly noted importance of OSS developers, the role of the Internet-based user community is introduced as fundamental in developing absorptive capacity for open source software development projects. We suggest that the Internet-based user community and the development group enhance the open source software project’s absorptive capacity by strengthening two salient capabilities: knowledge acquisition and knowledge transfer. Practical implications developed from these findings are presented.

Keywords: Absorptive capacity, open source software, software development

Introduction

The Internet is fundamentally altering the way in which software is developed and marketed. Not only does it facilitate the collaboration of developers from different organizations, cultures, and experience backgrounds on a software development project, it also broadens the ways in which users can participate in the development process. Previously, users were often co-located with developers and had a shared organizational affiliation. Internet-based user communities (IBUCs) comprised of a dynamic, diverse set of users, not recruited or selected by the organization, represent a significant departure from traditional user participation. Examples of such communities include those with a focus on proprietary software (http://forums.microsoft.com/) and open source software (OSS) (“SourceForge.net”).

OSS projects are on the forefront of leveraging the Internet to develop software, and there are many OSS projects that take advantage of diverse developer and user communities (see “SourceForge.net”). As an initial step towards understanding the antecedents of success for Internet-based collaborations, this paper focuses on OSS projects. Consistent with prior OSS research (Stewart et al. 2006), this study identifies OSS by the definition provided by the Open Source Initiative (see www.opensource.org). Research to understand the antecedents of success in OSS projects has focused primarily on the developers. Studies in this genre have examined developer motivation (e.g. Hertel et al. 2003), development team processes (Stewart and Gosain 2006), and developer license choice (Stewart et al. 2006). In this paper, we adopt a different lens to study the phenomenon of OSS development. Because
software development is a knowledge intensive activity, the importance of acquiring and using knowledge is critical to project outcomes. In the OSS context, relevant knowledge, such as the source code of other projects, is available to participants in all other projects, regardless of the trust or the strength of relationships between project participants. Because of such knowledge availability, the importance of social resources in determining access to knowledge is limited. Therefore, we focus on knowledge management and in particular draw on the absorptive capacity framework, to understand the antecedents of OSS development group performance.

Following prior research on OSS (Stewart and Gosain 2006), this study focuses on development group activity as an important indicator of performance for an OSS project. OSS development group activity is an important aspect of performance for OSS development groups because OSS developers are often volunteers. Thus, motivating development effort is an achievement for an OSS project. Further, it is a required prerequisite if the OSS development project is to achieve other goals such as software quality or software popularity.

The main contribution of this study is in the development and testing of a theoretical model explaining how the characteristics of IBUCs, developers, and their interaction facilitate knowledge acquisition, transfer and application to enhance performance. This paper extends the current OSS development research by highlighting and explaining the key role that users play in the OSS development process.

The remainder of this paper proceeds as follows. The next section summarizes prior work on absorptive capacity and highlights salient dimensions of absorptive capacity for OSS projects. This is followed by development of a model and hypotheses focusing on the role of the IBUC and development groups in enhancing absorptive capacity, and how that leads to performance. Following the research model, the methodology, descriptive statistics of preliminary data, and the current status of the project are presented. In the conclusion, implications of the results for practice and theory are discussed.

Absorptive Capacity

Drawing on the seminal work by Cohen and Levinthal (1990), we define absorptive capacity as an organization’s ability to recognize the value of new, external information, assimilate it, and apply it. Zahra and George (2002) defined two key dimensions of absorptive capacity: potential and realized absorptive capacity. Potential absorptive capacity makes an organization receptive to acquiring and assimilating external knowledge (Lane and Lubatkin 1998), while realized capacity is a function of the transformation and exploitation capabilities of an organization.

Potential absorptive capacity is critical for organizations that operate in highly dynamic industries. For such organizations, problems and solutions are continually changing, and so the organizations must be able to continually acquire knowledge to create solutions to relevant problems. Because technology changes rapidly, the technical needs of users also change rapidly (Iansiti and MacCormack 1997). As the technical needs change, the solutions change and organizations need to acquire knowledge about new solutions. Therefore, organizations seeking to develop software cannot rely on potentially outdated knowledge, suggesting a need for continuous knowledge acquisition and application. Thus potential absorptive capacity may be critical for OSS projects.

Researchers have suggested that the ability to acquire knowledge is enhanced when an organization has preexisting related knowledge (Cohen and Levinthal 1990; Fichman and Kemener 1999). Preexisting related knowledge is the extent of abstract knowledge, know how, and skills within the organization in areas related to the focal innovation (Fichman and Kemener 1999). Preexisting related knowledge is important because it provides the foundation necessary for understanding new knowledge and how the new knowledge is relevant to new innovations. For example, in an OSS development project, knowledge about HTML within a project enables that project to acquire knowledge about XML and understand why XML may be relevant to the OSS project.

There are two primary types of knowledge important to organizations that seek to innovate; knowledge of a problem that needs a solution and knowledge of the solution. Prior literature refers to these types of knowledge as awareness knowledge and how-to knowledge (Tornatzky and Fleischer 1990; Rogers 1995), respectively. We suggest that in OSS projects, these two distinct types of knowledge are located within two distinct types of participant communities.
**OSS Project Participants: The Developers and the IBUC Members**

Prior research has proposed classifications of participants in OSS projects. For instance, in their study of the GIMP project, an OSS development project that processes images in Linux, Ye and Kishida (2003) describe eight types of participants. These include the project leader, the core members, the active developers, the peripheral developers, bug fixers, bug reporters, readers and the passive users. However, while such granular categorizations of participants are appropriate for some of the largest OSS projects, they are less applicable to many of the smaller or less mature OSS projects.

We therefore consider a simpler categorization scheme and partition participants in an OSS project into two key roles; the developer and the IBUC member. The developers are those who actively develop source code and include Ye and Kishida’s project leader, core member and active developer roles. Then, drawing from the growing literature around OSS participant roles (e.g. Hertel et al. 2003), the IBUC for a given OSS development project is the set of participants who demonstrate an interest in the software developed by the OSS development project by participating in some computer-mediated communication related to the project, but do not regularly develop source code. The IBUC includes Ye and Kishida’s peripheral developers, bug fixers, bug reporters, readers, and the passive users.

The distinction between the OSS development group and the IBUC is important because these two participant types have different characteristics that may result in distinct knowledge acquisition capabilities. The developers, because they develop source code, may have a specific set of knowledge based on their understanding of the technical details of the project. Developers are therefore able to acquire some kinds of knowledge because they understand technical details about the project, or because the technical details of the project suggest that the new knowledge is relevant. While the IBUC is not expected to have knowledge of the technical details of the project, they may have other knowledge that facilitates OSS development group performance.

The IBUC may be important for knowledge acquisition for two primary reasons. First, IBUCs are likely to have significantly more members than the development group. For instance, von Krogh et al. (2003) note that for the OSS development project Freenet, 356 individuals participated on the discussion list, which indicates 356 participants in the IBUC, while there were only 30 developers. Likewise, in a study of Apache, Mockus et al. (2000) find that while about 400 people contributed source code, 3,000 people contributed by reporting problems with the software. Each IBUC member has access to some unique knowledge, thus having more IBUC members implies more unique knowledge available to the project.

Second, because the IBUC members may not share a programming background they may acquire knowledge that is different from the developers’ knowledge. For example, IBUC members are more likely than developers to acquire knowledge about problems encountered by users who are not technically proficient. Or, while developers are often experienced in software development (Zhao and Elbaum 2003), IBUC members may be experienced in other domains. IBUC member experiences in domains other than software development allows IBUC members to be better prepared, compared to developers, to acquire knowledge from business contexts other than software development. Such knowledge can lead to feature requests or bug reports that potentially are easily implemented by developers but that the developers may not have thought of because they do not have the business background necessary to understand the need for the feature. Consistent with Nambisan et al., we argue that innovation for OSS projects occurs at the “confluence of business expertise and technical mastery” (Nambisan et al. 1999).

When one subunit within an organization has better access to knowledge, as we suggest is the case with the IBUC in the OSS project, the ability of subunits to transfer knowledge is an important predictor of organizational performance (Argote, 1990, Hansen, 2002). Knowledge transfer is the process through which one unit is affected by the experience of another (Argote and Ingram, 2000). Because IBUC members may have greater access to know-what and developers specialize in know-how, a strong knowledge transfer capability is necessary to improve the effect of the IBUC knowledge acquisition on development group performance. (Argote and Beckman 1990; Hansen 2002). For this reason, we propose that a strong knowledge transfer capability is a critical dimension of absorptive capacity for OSS projects. In summary, we argue that the salient dimensions of absorptive capacity for OSS projects are IBUC and development group knowledge acquisition and project knowledge transfer. The overall research model underlying the study is shown in Figure 1. Below we develop specific hypotheses relating OSS knowledge acquisition and knowledge transfer to development group performance.
Hypotheses

Preexisting related knowledge is a key factor in determining an organization’s knowledge acquisition capability (Cohen and Levinthal 1990). We draw from the knowledge management literature (Cohen and Levinthal 1990; Hippel 1994) to identify two characteristics of preexisting related knowledge that are relevant for understanding the antecedents of OSS development group performance. The first is the diversity of preexisting relevant knowledge that is derived from participants’ diversity of experience. The second is the degree to which preexisting relevant knowledge comes from relationships with other OSS projects.

SourceForge Tenure Diversity

A diverse knowledge base is important for facilitating the acquisition of new knowledge for two reasons. First, the existing stock of knowledge held by an individual can facilitate the acquisition of certain types of knowledge, while inhibiting other types. For instance, at the individual level, knowing algebra may facilitate the acquisition of calculus knowledge, but not knowledge about historical events (Ellis 1965). At the organizational level, a diverse knowledge base facilitates an organization’s ability to acquire diverse knowledge. Acquisition of diverse knowledge is important for a software development group because innovation in general, and software development in particular, improves when different perspectives meet (Cohen and Levinthal 1990). Specifically, innovation often occurs by connecting knowledge that may have previously been considered unrelated and reinventing the knowledge in a new context (Cohen and Levinthal 1990).

Individuals who are diverse based on their tenure, or time spent, with a particular culture are likely to have diverse knowledge and thus a stronger ability to acquire diverse knowledge. Sourceforge is one of the most widely used development platforms, and a large community of users and developers interact through that platform. Time spent interacting in the culture which is associated with SourceForge may represent an IBUC member’s time spent with OSS development and OSS applications. Long-term SourceForge use implies that the participant has been using or developing OSS applications for a long time, and that length of time with OSS applications may have facilitated a certain software related knowledge base. In contrast, short-term SourceForge participants may have a different set of software related knowledge, based on experiences with proprietary software. Therefore expectations about how software should work or be developed may be different depending on the time a participant has been involved with SourceForge. Having distinct knowledge bases, some based on long histories with SourceForge and others reflecting shorter histories is expected to lead to a larger set of unique knowledge that yields a stronger knowledge acquisition capability.

IBUC SourceForge Tenure Diversity

Desired functionality and ease of use are key topics on which users are well positioned to acquire knowledge. IBUC members are therefore expected to make contributions related to these topics. IBUC members who have been on SourceForge longer should be experienced in the SourceForge development and distribution process. In particular they should be experienced with how to acquire knowledge that will lead to actionable feature requests and bug reports. However, IBUC members who have been on SourceForge for a long time may also be constrained by their experience with SourceForge. They may have difficulties acquiring knowledge that does not fit into the SourceForge development and distribution process. IBUC members who are new to SourceForge may be more likely to acquire knowledge that is distinct from that which can be acquired by IBUC members who have been on SourceForge for a longer period of time. IBUC members with a shorter tenure on SourceForge can acquire knowledge that may not immediately fit well into the SourceForge software development paradigm. The combination of the short-term and long-term IBUC members is likely to lead to a stronger knowledge acquisition capability because they can acquire a larger set of unique knowledge. In addition, having both short and long term IBUC members can have a synergistic effect. The long-term IBUC members may be able to help the short-term IBUC members create actionable requests based on their acquisition of innovative knowledge.

In summary, IBUC members play a significant role by acquiring knowledge about desired functionality and features of the software. A broader range of tenure with SourceForge in the IBUC is likely to lead to a larger base of knowledge through which to acquire new knowledge. Furthermore, a wider range of tenure is expected to enable a synergy where IBUC members who have been on SourceForge longer can guide the innovative ideas of the shorter
tenure IBUC members. Acquiring relevant new knowledge and combining it in ways that can be leveraged to enhance the software will have a positive effect on the overall group project performance. We therefore propose:

**Hypothesis 1:** As IBUC tenure diversity increases, development group performance will increase.

**Development Group SourceForge Tenure Diversity**

Although IBUC SourceForge tenure diversity is important for development group performance, the development group SourceForge tenure diversity is also expected to be an important facilitator of knowledge acquisition. While the IBUC’s influence is primarily through knowledge acquisition about desired functionality and ease of use, the development group SourceForge tenure diversity is expected to have an effect through knowledge acquisition related to techniques for software development.

Experience as a developer is of considerable value. Experienced developers may be better able to work with others in a group and have refined development skills. The knowledge of experienced programmers may allow them to acquire knowledge of complex or elegant programming solutions. However experience as a developer, as signaled by tenure on SourceForge, may also be limiting. Experience with SourceForge could constrain the developers’ ability to acquire knowledge. A more tenured SourceForge developer may be biased toward the development styles available on SourceForge and therefore have trouble identifying the relevance of new development styles. Therefore it may be difficult for developers with longer tenures on SourceForge to acquire knowledge about other potentially useful development styles, languages or programming techniques. By contrast, developers with shorter tenures with SourceForge may be more open to and prepared to acquire this kind of knowledge.

Shorter tenures with SourceForge may also signal less experience as a programmer. Less experienced programmers may be familiar with a different set of programming languages and techniques that allow them to acquire knowledge about these techniques. As an example, a more experienced programmer with a strong foundation in C++ may not be as prepared to acquire knowledge about the newest Java advancements as a less experienced programmer who knows Java. In summary, development group SourceForge tenure diversity is expected to signal a more diverse source code development related knowledge base. This more diverse knowledge base should then lead to more options for how to find and fix bugs and find and add features. Further, as in the case of the IBUC, there may be a synergy between the development expertise of the long-term SourceForge developers and the innovative ideas of the short-term developers. The short-term developers can offer ideas for development and the long-term developers can help implement them. We therefore propose:

**Hypothesis 2:** As development group tenure diversity increases, development group performance will increase.

**Project Relationships**

The organizational product development literature notes that relationships with other organizations are an important source of external knowledge (Cohen and Levinthal 1990; Hippel 1994). Through its connections to other organizations developing similar products, the focal organization can gather important knowledge about success and failure in the industry. Therefore we expect the knowledge acquisition capability reflected by relationships with other OSS projects to lead to increased performance. A relationship with another OSS project is defined by an IBUC member or developer on the focal project participating in computer-mediated communication with another OSS project on SourceForge.

**IBUC Project Relationships**

A large number of relationships with other OSS projects is suggestive of increased IBUC member familiarity with OSS applications. IBUC member familiarity with other OSS applications may enhance knowledge acquisition that will result in development group performance for two reasons. First, IBUC member relationships enable IBUC members to acquire knowledge about potential features that are easier for developers to implement than features that IBUC members learn about from use of proprietary software. When an IBUC member requests a feature that is implemented in another OSS project, it may be easier for developers to implement because the developer can re-use the code from the original project. For instance, an IBUC member of the ZK project suggested that the software write “not just to Dhtml but to Flash” and that developers could look at code from OpenLaszlo, another project on SourceForge.net, to find an example of how to write to Flash. Because IBUC member knowledge from
relationships with other OSS projects facilitates the addition of features, IBUC member knowledge from relationships with other OSS projects facilitates development group performance.

Second, IBUC member relationships with other OSS projects enables knowledge acquisition about other projects that may result in unique feature suggestions about collaborations between two OSS projects. Because the set of OSS projects that IBUC members are related to may be driven more by context they can provide knowledge about collaborations between OSS projects based on context solutions. For instance instead of being related to projects that all use the Java development language, as developers may be, IBUC members may be related to OSS projects that provide solutions relevant to researchers. IBUC members are thus better able to suggest features that enable software compatibility between software that researchers use. For these reasons we propose:

Hypothesis 3: As the number of IBUC project relationships increases, development group performance will increase.

Development Group Project Relationships

Relationships between developers in different OSS projects have benefits similar to those suggested by networks of practice. For instance, the literature focused on boundary spanners provides evidence that those who perform the same job, across not just organizations but also across shifts in a manufacturing facility may increase performance (Epple et al. 1991) because of the knowledge transfer that occurs. In particular, for OSS projects, developer familiarity with other OSS projects may lead to the acquisition of knowledge concerning success and failures related to adoption of programming styles or opportunities to gain resources. Knowledge from other OSS projects is likely to be especially relevant because the source code is accessible, which may offer insights into how to add features and correct bugs.

In addition, developer relationships with other projects may signal knowledge that the developers have about skills possessed by developers on other projects. Knowledge of who knows what is important for networked organizations. This knowledge may be used to recruit developers to the project to add features. In this way developer relationships may lead to development group performance. We therefore propose:

Hypothesis 4: As the number of development group project relationships increases, development group performance will increase.

Knowledge Transfer Capability

Although the IBUC and the development group each enable knowledge acquisition and thereby enhance project performance, the relationship between the IBUC knowledge acquisition capability and development group performance should be stronger for projects with a strong knowledge transfer capability. Knowledge transfer is facilitated by repeated communication (Uzzi 1997) that facilitates an exchange of perspectives. Repeated communication allows the IBUC members and the developers to share knowledge so that their shared knowledge base increases and knowledge transfer is easier and more effective. Further, as more developers interact with the IBUC, the IBUC will begin to understand the kinds of tasks that the development group can and will implement and therefore be better able to make useful requests. Interaction between IBUC members and developers is evident in communication threads on project forums where both participate. We therefore propose:

Hypothesis 5: The number of threads posted in forums related to the OSS project that include both IBUC members and developers will positively moderate the relationship between IBUC tenure diversity and development group performance.

Hypothesis 6: The number of threads posted in forums related to the OSS project that include both IBUC members and developers will positively moderate the relationship between the number of IBUC project relationships and development group performance.
Methodology

The development and effects of absorptive capacity for OSS projects will be explored using data from a sample of OSS projects hosted by SourceForge. SourceForge is one of the largest repositories of OSS projects. The focus of this research is on understanding what leads to success and not how successful projects function, therefore a selection of projects that are early in their lifecycle is most appropriate. Projects for the sample will be chosen based on the project’s registration date with SourceForge, the project’s use of public forums, the project’s development group activity, the project’s application domain and the project’s use of the GNU General Public License (GPL). By choosing projects that have similar registration dates, we limit the project variation that may be due to the time that the project was registered on SourceForge. We focus on projects that use public forums because a central aspect of this study is in understanding knowledge transfer. If there is no knowledge transfer in public forums the project may use another form of computer-mediated communication such as mailing lists. Because mailing lists were not observed this would confound the results. SourceForge is used to distribute software and to develop software, so screening for projects with a minimum level of development group activity will limit the sample to projects that were under development during the time frame considered. Prior research suggests that software that is developed for the use of developers attract more activity than software that is designed specifically for use by non-developers; thus application domain is controlled for in the analysis (Fershtman and Gandal 2004). Finally, the GPL is one of the most popular OSS licenses; therefore this restriction ensures that the projects are OSS projects according the OSI initiative. By selecting projects that use the same license, variation in projects that prior literature suggests may be related to the license used is limited (Stewart et al. 2006).

The variables in the model will be operationalized using a database supplied by SourceForge to the University of Notre Dame. The IBUC for each OSS project is identified using ids for individuals that participated in some computer mediated communication with the project, such as reporting a bug or posting to a forum, during the first year after the project was registered. Those listed as developers or administrators for the OSS project are excluded from being a part of the IBUC and are identified as developers.

The IBUC SourceForge tenure diversity is operationalized as the variance of the number of days since each IBUC participant was registered with SourceForge. The number of relationships with other OSS projects for the IBUC is
the sum of the relationships with other OSS projects across all IBUC members. A relationship is identified by an IBUC member participating in some computer-mediated communication with a project other than the focal project. The measures for the development group are similarly counted for developers.

To measure the knowledge transfer variable, the public forum(s) on SourceForge for each OSS project are examined. Each thread that contains a post from both a developer and an IBUC member during the first year after the project was posted on SourceForge is counted.

The dependent variable, development group performance, is measured as the percentage of tasks closed out of those available to be closed during the project’s second year after registration on SourceForge. Tasks include features requests, support requests and bugs reports. The percentage of tasks closed is consistent with the task completion measure used by Stewart and Gosain (2006) to measure OSS development project success. In addition to the percentage of tasks closed, the number of lines of code added during period two is used to measure the developer group performance. Linear regressions are applied to explore the relationships proposed. The number of developers and the number of members of the IBUC represent controls in the model.

**Status and Future Plans**

Pilot data has been collected for 78 OSS projects that fit the requirements described above. Preliminary descriptive statistics for these projects are included in Table 1. During the next few months the sample will be expanded to include at least 100 projects, the variables described will be calculated for each project, and the model proposed will be tested. Results will be available for presentation in December.

<table>
<thead>
<tr>
<th>Table 1. Descriptive Statistics</th>
</tr>
</thead>
<tbody>
<tr>
<td>Min</td>
</tr>
<tr>
<td>Number of Developers</td>
</tr>
<tr>
<td>Number of IBUC Members</td>
</tr>
<tr>
<td>Number of Threads</td>
</tr>
</tbody>
</table>

Conclusion

Despite the emergence of IBUCs in OSS development and other Internet-based organizations, research to understand how IBUCs contribute to these organizations’ performance is limited. This paper takes an initial step towards understanding the effects of both the IBUC and development group characteristics and behaviors on innovative organizational outcomes. Specifically, by focusing on participant characteristics and behaviors this research seeks to inform managers concerning the type of IBUC and development group members to target for participation. Building an IBUC that has a positive influence on development should be important to managers of proprietary software and other innovative organizations. An IBUC provides a great opportunity for proprietary software to potentially enter new markets by getting feedback from users in a variety of business contexts. Finally, by highlighting the impact of an IBUC, this paper contributes to the growing literature that seeks to understand the factors that lead to success for OSS projects.

While this research focuses on the influence of IBUCs early in the development life-cycle of OSS projects, there are many ways in which this research can be extended. Future research should be conducted to understand the characteristics and behaviors of participants that lead to maintaining the most popular OSS products such as Linux and Apache. In addition, while this study focuses on exploring the relationships between theoretically driven characteristics and behaviors, this research can usefully be complemented by a qualitative study that may reveal other characteristics and behaviors, such as the disciplinary background of developers, that lead to internet based innovation.
References


