Emergence of New Project Teams from Open Source Software Developer Networks: Impact of Prior Collaboration Ties

Jungpil Hahn†, Jae Yun Moon‡, and Chen Zhang○

Last Revised: January 14, 2008

ABSTRACT

Recent years have witnessed a surge in self-organizing voluntary teams collaborating online to produce goods and services. Motivated by this phenomenon, this research investigates how these teams are formed and how individuals make decisions about which teams to join in the context of open source software development (OSSD). The focus of this paper is to explore how the collaborative network affects developers’ choice of newly-initiated OSS projects to participate in. More specifically, by analyzing software project data from real world OSSD projects, we empirically test the impact of past collaborative ties with and perceived status of project members in the network on the self-assembly of OSSD teams. Overall, we find that a developer is more likely to join a project when he has strong collaborative ties with its initiator. We also find that perceived status of the non-initiator members of a project influences its probability of attracting developers. We discuss the implications of our results with respect to self-organizing teams and open source software development.

Keywords: Open source software development (OSSD), team formation, developer social networks, collaborative ties

† Krannert Graduate School of Management, Purdue University, West Lafayette IN, 47907. (1-765-494-2188, jphahn@mgmt.purdue.edu).
‡ Business School, Hong Kong University of Science and Technology, Clear Water Bay, Kowloon, Hong Kong. (852-2358-7635, jmoon@ust.hk).
○ Fogelman College of Business and Economics, The University of Memphis, Memphis, TN 38152. (1-901-678-5671, C.Zhang@memphis.edu).
1. INTRODUCTION

Recent years have witnessed the emergence of a new form of producing information and cultural goods. Enabled by the pervasive digitally networked environment, individuals distributed worldwide are collaborating to produce goods and services ranging from software to free encyclopedias (Benkler 2006). This new mode of production is characterized by self-organizing volunteer groups formed through virtual interactions that occur largely over the Internet and has also made possible new forms of collaborative partnerships between businesses and customers on an unprecedented scale (Tapscott and Williams 2006). Today, an increasing number of companies are foraying into establishing relationships with users in recognition of the value-creating potential of these voluntary groups. In order to gauge the effectiveness and potential impact of these groups, it is important to understand how these volunteer groups form and evolve as individuals sign up and participate in them. Individuals’ decisions about which groups to join have an impact on the resultant aggregate group composition, which has been shown to be an important determinant of group performance through its impact on group cohesion and coordination (Beal et al. 2003, Gruenfeld et al. 1996, Levine and Moreland 1990). Group formation is a pervasive social phenomenon and the mechanisms through which groups are formed have been a subject of theoretical interest for academics in diverse fields (e.g., Backstrom et al. 2006, Ruef et al. 2003).

In this paper, we study how voluntary software project teams emerge from the social networks within which they are embedded in the context of open source software development (OSSD). While OSSD is not a new phenomenon, the business interest in this phenomenon has only recently surged as the prominence garnered by well-known OSSD projects such as the Apache Web Server and the Linux operating system kernel served as testimonies to the attractiveness and viability of OSSD as an alternative to the conventional proprietary model of producing software (O'Reilly 1999, Raymond 2001). This context is chosen both because it is representative of a broader general phenomenon of theoretical interest as well as having practical significance in understanding team formation mechanisms that have the potential to impact OSSD success. Theoretically, the study of OSSD project team formation enables us to examine how the existing web of relationships will affect new team emergence in an environment where
the structure of the existing network and the resultant new teams are visible. Just as the social position of an individual within a network of peers influences his career advancement opportunities (Burt 1992), or as the social position of a firm within a network of organizations influences its alliance strategies and subsequent outcomes (Gulati 1995, Powell et al. 1996), the composition of an OSSD project team in terms of its developers and their positions vis-à-vis the OSSD network may influence its technical and commercial success through its impact on the likelihood of attracting developer attention (Grewal et al. 2006). Because the position of a developer in the OSSD network is a result of joint participation with other developers in past projects, we theorize that past collaborations in OSSD will impact how new project teams take form. As a case in point, the original Linux operating system kernel development group was not formed out of a social vacuum. Linus Torvalds, the project initiator, had been an active member in a related community of minix programmers and it was out of this community that the first volunteer developers emerged to form the Linux kernel developer team (Moon and Sproull 2002). A recent study that examined how sub-projects emerged within the Apache Web Server project found that most included at least a few developers who had worked together previously on other Apache sub-projects (Weiss et al. 2006). In short, the OSSD community can be regarded as a dynamic collaborative network of developers whose structure of connections shapes and is reshaped through the formation of new project teams. The resultant structure in turn influences project outcomes through its impact on access to developers and critical resources embedded in the network. Thus, project team emergence from OSSD communities is a natural context for studying how individuals are influenced by the network within which they are embedded. However, despite the relevance and importance of the collaborative network in OSSD, few studies have examined its antecedents and impact on project team formation mechanisms.

Practically, understanding the mechanisms of team formation will have implications for the increasing number of OSSD projects launched by both commercial and non-commercial actors, the vast majority of which fail to take off and become abandoned (Chengalur-Smith and Sidorova 2003). First, unlike organizational software development teams that are usually formed by managers who consider
developer skills and experiences, an OSSD project team must rely on developers voluntarily choosing to become members of the project (Grewal et al. 2006, von Hippel and von Krogh 2003). Furthermore, OSS project success can be measured through the extent to which it attracts and sustains volunteer developer interest and active contributions (Crowston et al. 2006). While there has been considerable research examining developer motivations to participate in OSSD, in general, despite the lack of monetary compensation (e.g., Hars and Ou 2002, Hertel et al. 2003, Lerner and Tirole 2002, Roberts et al. 2006), there has been little research that examines developer decisions as to which project to join when provided with similar alternatives. Examining developers’ particular choices can provide a deeper understanding of developer motivations than when one examines general attitudes toward whether or not to participate in OSSD. Recent research has begun to examine how factors such as the restrictiveness of the software distribution license and nature of organizational sponsorship may interact in influencing user interest and developer motivations to contribute to established OSSD projects (Stewart et al. 2006). There is a paucity of research however, that examines what determines which new projects developers join that project initiators may draw upon in order to better understand how to increase their chances of attracting developers. In this paper, we focus on such new project joining behaviors due to theoretical, practical and methodological concerns. Theoretically, although examining established teams, as has been the case of most prior studies of team formation, can help us understand how and why new members decide to join such teams, little can be said about how those teams were created and became established in the first place. Focusing on newly created projects would enable us to study team formation from a new light without the possible confounding effects of the characteristics of established teams. In terms of practice, because early marshalling of developer interest is critical for ensuring sustainability of OSS projects (Raymond 2001), examining developer choice of newly-initiated projects will not only contribute to a deeper understanding of the ecology of OSSD projects but also help devise practical guidelines for project managers to increase project attractiveness to developers. Furthermore, member composition can also affect software development performance through its impact on administrative and expertise coordination effectiveness (Faraj and Sproull 2000). Software development performance is also affected by the
programming skills and application domain experiences that the team members bring to the project (Boehm 1987, Curtis et al. 1988). Therefore, from a practical standpoint, understanding new project team formation may provide insights on key problems related to early staffing of new OSSD projects and subsequent performance. Finally, in terms of research methodology, focusing on new projects enables us to construct a unique dataset of team formation with (almost) continuous tracking of developer joining events. This provides richer and more accurate insight into the team formation dynamics than would be possible with coarse-grained archival data of established projects where the sequence of member joining is imprecise, resulting in inaccurate inferences regarding the factors influencing joining decisions.

The remainder of this paper is organized as follows. In the next section, we present our theoretical background and develop our research hypotheses. Next we outline our methods and present the results. We conclude by discussing the implications, contributions and directions for future research.

2. THEORETICAL FRAMEWORK AND HYPOTHESES

2.1. A Social Network Perspective of Open Source Software Development

The OSSD community can be understood as a collaborative network of developers working together in different project teams. Developers participate in one or more OSSD project teams and in doing so develop ties with other team members. The social nature of development activities in OSSD where developers and users form a complex network of relationships via various electronic communication channels on the Internet (von Hippel and von Krogh 2003) has spawned a set of studies examining this phenomenon using a social network perspective. Madey, Freeh, and Tynan (2002) conducted one of the first empirical investigations of the open source movement from the social network perspective and found that the OSSD community exhibits properties of self-organizing networks. Others have proposed methodological applications of social network analysis to data obtained from concurrent versions systems (CVS) code repositories of OSS projects (Lopez-Fernandez et al. 2004). Xu, Gao, Christley, and Madey (2005) explored social network properties in the open source community to identify patterns of collaborations. While early work from the social network perspective has mostly attempted to provide descriptive accounts of the network properties of the OSSD community, more recent studies have
started to theorize the impact of social relations on OSSD outcomes. Grewal et al. (2006), for example, found that the network embeddedness of OSSD projects and their managers vary largely across a set of projects that have adopted the Perl programming language as the platform technology, and that this variation resulted in differential success rates. We contribute to this growing stream of studies by examining why and how the structure of existing collaborative ties in OSSD networks impact individual developer decisions regarding the choice of new projects to join, which in turn may result in the formation of new collaborative ties.

Because of this focus on collaborative ties, we also adopt a social network perspective. Social network analysis aims to understand the implications of the relationships between people, groups, organizations, and other types of social entities (Granovetter 1973, Wasserman and Galaskiewicz 1994, Wellman and Berkowitz 1998). A social network is modeled as a graph with nodes representing the individual actors (i.e., OSSD developers) in the network and arcs representing the relationships or ties between the actors (i.e., collaboration in an OSSD project). In a social network, the actors develop and maintain a tie by exchanging either tangible or intangible resources such as goods, services and information. In the OSSD network, the developers form ties by virtue of collaborating with other developers on software projects. The ties may vary in strength depending on a number of factors such as the amount of time, the emotional intensity, the intimacy, and the reciprocal services associated with the relationship (Granovetter 1973). Strong ties are characterized by a sense of special relationship, an interest in frequent interactions, and a sense of mutuality of the relationship (Walker et al. 1994) whereas weak ties are maintained infrequently or indirectly between the actors who belong to different social clusters. Both strong ties and weak ties play an important but differential role in a social network. Strong ties maintain and promote trust and collaboration whereas weak ties enable actors to access resources and information that may be unavailable in their immediate social circles (Burt 1992, Granovetter 1973).

In this paper, we focus on prior collaborative ties among developers as a potential driver behind developer joining behavior and project team formation; in short, we examine how collaborative networks impact the project teams that emerge. When developers decide to join a project, in addition to factors
regarding the use value of the project (Hertel et al. 2003, Roberts et al. 2006) and expected enjoyment and
learning benefits (Hars and Ou 2002, Lakhani and Wolf 2005), they may also consider factors that are
related to the composition of the project team as this may impact how the development process will
unfold. For example, a developer may be concerned about issues related to coordination and
communication with other team members since much of the collaboration occurs asynchronously over
computer-mediated distributed networks. In general, people prefer to work with those with whom they
have worked in the past because of the reduced uncertainty stemming from familiarity based on prior
collaborative experiences (Hinds et al. 2000). In addition, the social capital that the project’s existing
developers have developed by virtue of having collaborated with other developers in the overall OSSD
network can affect project outcome through access to information and other resources critical for effective
software development (e.g., Grewal et al. 2006), and may serve as a potential source of reputation
enhancement by association for joining developers (Stewart 2005).

2.2. Perspectives on How Prior Collaborative Ties Affect Project Team Formation

In this section we integrate emergent findings from research on motivations of OSS developers
with theoretical perspectives derived from organizational behavior, social psychology and sociology that
have examined the mechanisms determining the formation of naturally occurring groups both within and
outside organizations (e.g., Ruef et al. 2003). Conventionally, project teams in organizations are
strategically formed by a manager who assigns individuals to a team based on characteristics such as
expertise and personality. Alternatively team members may self-select into teams. Similarly, in OSSD,
projects may recruit developers both formally (e.g., by broadcasting position openings and required
qualifications to the OSS community),¹ and informally (e.g., by inviting other developers to join) or
alternatively developers may voluntarily ask to join a project team.

¹ Interestingly, the extent of formal recruiting is surprisingly low (von Hippel and von Krogh 2003). For example, there are only about 200 position openings posted on SourceForge.net at any given time. This figure is quite inconsequential when we consider that there are over 150,000 projects of which several thousand have substantial development activity. For example, we observe over 3,000 projects to have had at least 10 code commits between June 1 and June 15, 2007.
Research on group formation – be it work or social, self-organized or prescribed – indicates that group formation is a result of the deliberate, strategic decisions of individuals who either self-select or assign others to a group with the purpose of satisfying individual and group objectives (Owens et al. 1998). In naturally forming groups, the process is typically bi-directional – the individual seeks to find and join a group that will satisfy his goals and needs, the group seeks individuals that fit and fulfill its goals (Levine and Moreland 1990). Similarly, in the OSSD context, developers are motivated to choose projects and collaborators that can afford them ample opportunities to realize the intended benefits of participation; projects need developers with the required skills and experience to move development forward. At the outset we acknowledge that both developers and projects may evaluate the other in the process of project team formation (i.e., either developers request to join and project administrators decide whether or not to accept them, or project members invite developers who then decide whether or not to join), but we find little evidence of project administrators either rejecting developer requests to join or actively inviting developers to join in the context of OSSD projects. Hence, we frame the remainder of our discussion in terms of developer self-selection into projects.

Developer participation in OSSD is driven by motivations ranging from enjoyment and learning from development activities, a sense of obligation toward the OSS community, a belief that code should be open, a need for the software developed, to a desire for reputation (Lakhani and Wolf 2005). Maximal realization of these goals depends on the success of projects to which they contribute, both in terms of outcome (i.e., sustained successful development of code), as well as process (i.e., how well project members work together). The former is important if one is to fulfill the need for software as well as give back to the OSS community; the latter will likely influence the degree to which participants will learn and enjoy participating in OSSD. However, from the perspective of prospective developers, it is difficult to

---

2 We surveyed the administrators of our study sample projects (\(N = 384\); response rate = 16.3%). Of 384 project initiators, only 48 (12.5%) stated that they were selective in accepting new developers to the project team. Most administrators were receptive of developer join requests, a tendency well illustrated by some of our respondents’ comments: “volunteers are almost always welcome”, “everyone should be allowed to participate in open source”. Only 15.6% of project initiators invited developers to join the project.
predict the likelihood of future project success because of the inherent uncertainty in judging the true objective quality of the project and the programming skills, project administration skills, and work styles of current project team members. In order to deal with this uncertainty, developers will rely on other cues and attributes as proxies of both the project and its team members’ underlying quality. Prior work on self-organizing networks suggests that people typically rely on two social cues – cohesion and status – to determine the likelihood of successful collaboration when the quality of a future collaborator is difficult to measure directly (Guimerà et al. 2005, Uzzi et al. 2007). Cohesion is related to preference for repeat collaborations in order to benefit from close work relationships; status is related to preference for collaborations with experienced, well-established actors in the network. Past collaborative ties play a critical role in shaping both cohesion and status cues. Cohesion cues result from direct past ties between the prospective developer and existing project members; status cues result from the past collaborative ties of existing project members with the broader OSS developer network. Developers rely on cohesion cues because they are motivated to continue to work with those with whom they have successfully collaborated in the past. Cohesive ties from past interactions can result in greater trust and knowledge of the technical and organizational skills possessed by the potential collaborators (Uzzi and Spiro 2005), and hence influence the developer’s perception of the likelihood of project success – both in terms of the collaboration process and outcome of the project. Developers rely on status cues because they are motivated to work with those whom they perceive to be successful. Status of existing project members as observed through their ties in the network can signal their expertise and quality as vetted by developers in the overall OSSD network, and hence affect the prospective developer’s perception of the likelihood that the project will be successful. Thus, we propose that past collaborative ties of the existing developers – with the potential joiner and with other developers in the OSSD network – will influence developers’ decisions to join the project. We explore how collaborative ties relate to these two types of cues, and as a consequence affect project team formation in more detail below.
2.2.1. Cohesion Cues: Prior Collaborative Ties between Developer and Existing Project Members

Research suggests that people are more likely to work together when they have prior social ties (McClelland et al. 1953, Schachter 1959) and that past collaborative ties tend to be repeated in the future (e.g. Kogut 1989, Uzzi and Spiro 2005). When choosing future collaborators, a developer may rely on ties developed through joint participation in past projects as a social cue to assess their underlying quality and the risks involved in working with them. Prior collaborations help him obtain information about the skills and capabilities possessed by others (Granovetter 1985, Coleman 1990). Hence, he can better rely on personal experiences with the existing project members to judge the likelihood of the successful outcome of the new project. Past experiences with the project members may also help the developer better estimate the risk and uncertainty involved in the collaboration process. Teams consisting of individuals with preexisting relationships have been shown to solve complex problems better than teams of strangers because they are able to pool information more efficiently (Gruenfeld et al. 1996). Software development teams composed of members with prior joint project experience may be more effective in coordinating programmers’ distributed expertise because they have developed knowledge of ‘who knows what’ (Moreland 1999, Faraj and Sproull 2000). In the OSSD context in particular, due to the lack of opportunities for face-to-face contact, developers face greater barriers to effective communication and coordination and are thus more likely to be concerned about these issues (Kotlarsky and Oshri 2005). Therefore, developers will be more likely to repeat interactions with developers whom they have worked with in the past, thereby reinforcing collaborative ties. As developers interact more frequently, the strength of the collaborative tie will increase as they develop closer and more cohesive working relationships (Granovetter 1973, Hansen 1999), further increasing the likelihood that they will collaborate in the future. Thus, we hypothesize:

H1: The likelihood that a developer will join a project is positively related to the strength of his collaborative tie with the project’s existing members.

Not all past collaborative experiences are necessarily successful and many projects fail due to conflict between project members, among other factors. Research on group formation in laboratory and
field settings has found that people are attracted to groups when their prior experiences with key group members have been positive and successful (Hinds et al. 2000, Zander and Havelin 1960) and that successful outcomes sustain group cohesion (Boone et al. 1997). If developers are motivated to maximize the chance that they gain the expected benefits from OSS project participation, then they will be more likely to choose the same collaborator if the past experience with him was satisfactory not only in terms of the quality of the collaboration process but also in terms of final project outcome. In addition, positive collaboration experiences may have created implicit obligations for future exchange due to the benefits that the developer derived from association with the project members in the past, in particular if the past collaboration experience was one in which the current project members contributed substantially to the developer’s past project. Thus, we hypothesize:

H2: The likelihood that a developer will join a project is positively related to the outcome of his past collaborations with the project’s members.

2.2.2. Status Cues: Prior Collaborative Ties between Existing Project Members and OSSD Network

In addition to knowledge and impressions formed through direct prior collaborations with the project members, a developer may also rely on status cues, other observable proxies to judge the quality of the prospective collaborators, in particular in the absence of direct past contact. Here status represents an individual's prestige or honor in the social network (Weber 1968). Social scientists have argued that the perceived status of an individual is associated with his relationship with others (Frank 1985) and that status can be based on the number of ties he has developed (Podolny 1993) or his position in the social network (Stewart 2005). Prior literature in sociology has examined patterns in alliance formations and collaborative relation formation and found that high status actors with many connections in the network are more likely to attract partners than less connected actors (Moody 2004, Podolny 1993, Powell et al. 2005). In the OSSD context, the number of connections through past project participation that existing project team members have amassed may signal to others their ability for valuable contributions to OSSD projects. High status (i.e., highly connected) developers may have gained a vast store of experience
through their past collaborations with others in the network, and thus will be perceived to be more competent and more likely to succeed (Stewart 2005, Thye 2000). In addition to serving as a proxy of the underlying quality of project members, the number of prior collaborative ties may also provide the project with greater access to other OSSD network participants and their resources and hence increase the likelihood that the project will succeed. Therefore, in OSSD, we propose that developers will prefer participating in projects whose existing members are perceived to have higher status based on their prior collaboration experiences in the OSSD network. Thus, we hypothesize

**H3:** The likelihood that a developer joins a project is positively related to the perceived status of the project members in the open source software developer network.

We tested our hypotheses empirically using data collected on a sample of newly initiated projects from an online project management platform for OSSD projects. We describe the data collection process, measures and analytical procedures in the following section.

3. DATA AND METHODS

3.1. Study Sample

Data was collected from SourceForge.net, the largest repository of OSS projects on the Internet. At the time of data collection, SourceForge.net provided free hosting to more than 150,000 projects and more than 1,600,000 subscribers.³ It also offers a variety of services to hosted projects such as mailing lists, bug trackers, message boards, file repositories, code CVS, and other project management tools. SourceForge.net has been an attractive source of data for many OSS researchers mainly due to the abundance of publicly accessible data (Howison and Crowston 2004).

We selected all public OSSD projects newly registered at SourceForge.net between September 30 and November 11, 2005 (N = 2349) as our sample of projects. A software crawler visited these projects’ web pages hosted at SourceForge.net and kept track of project-related and membership information on a daily basis. This process not only enabled us to distinguish between the initiator and the developers who

³ Howison and Crowston (2004) and Rainer and Gale (2005) note however that many of these projects are inactive.
subsequently joined but also enabled us to capture the timing (i.e., the sequencing) of joining events as well as timing of important project events (e.g., release of code, updating of project description). Daily tracking ended on January 5, 2006.

At SourceForge.net a developer interested in starting a new project submits a request to the SourceForge.net staff. After the project is approved for hosting, the project initiator can start utilizing the services and tools provided by the site and upload contents to the project site. A developer wishing to become a member of the project first contacts the project administrator, who then either approves the joining request and adds him to the project membership or rejects the request. A joining event occurs when a developer’s joining request is approved by the project administrator and the developer becomes a listed member of the project. A project becomes a team only after at least one developer joins the project. In other words, we distinguish between single-developer projects (i.e., the initiator is the only member) and multi-developer project teams (i.e., when additional developers have joined the project) and use the term team only when we refer to the latter.

Our data also includes all the developers who had participated in at least one project hosted at SourceForge.net prior to the new project data collection period ($N = 170,741$). We selected developers with project experience for two reasons. First, developers without prior project experience are unlikely to have developed the collaborative ties with other developers that are the focus of our study. Second,

---

4 Because SourceForge.net is set up in such a way that only one user can “officially” register a project, even if a project is jointly founded by two or more developers, the data would show that only one user is the initiator and the others are developers that subsequently joined, presumably very early or right after initiation. Hence, treating the other co-founders as external developers who joined the project after its initiation may lead to an underestimated impact of collaborative ties. Such co-founding situations would materialize in the dataset as a project initiation followed by a very rapid joining of a developer with ties to the initiator. In our sample we identified 20 (of 1198) joining events where developers joined within the first week and the joining developers had past collaborative ties with the initiators (i.e., those likely to be co-founders) and discarded them from further analysis.

5 While project administrators may invite other developers to join or sometimes reject developer requests to join the project, a survey of our study sample indicated that this occurs infrequently (see footnote 2). Consequently, we frame our discussion on developers requesting to join, which seems to be the norm in OSSD.
including the developers with past project experience enables us to operationalize measures of fit between developers’ technical profiles and project technical requirements.6

Developers’ collaborative ties were identified using data about projects and developers from the repository of SourceForge.net’s project databases hosted at the University of Notre Dame (http://www.nd.edu/~oss/). Developers’ past collaborators were identified through their co-memberships on other OSS projects. Based on this data, we constructed affiliation matrices of developers and projects that depict the existence of prior collaborative relationships between developers.

In summary, the final dataset used to construct our sample consists of 2349 projects and 170,741 developers. The unit of analysis is a triad consisting of a date, a project, and a developer – (T, P, D), in which T is a date between September 30, 2005 and January 5, 2006, P is one of 2349 projects, and D is one of 170,741 developers.

3.2. Measures

**Dependent Variable**

The final outcome of interest for a triad (T, P, D) is whether developer D joins project P on date T, which is coded as a binary variable (Join).

**Independent Variables**

**Cohesion Cues: Strength and Outcome of Collaborative Ties.** Measures of collaboration outcomes and the strength of collaborative ties between developers were computed based on developers’ project histories. We defined past collaborators as those developers who had been concurrently listed as members of a common project prior to the focal date. In other words, developers who have been involved in the same project at the same time are coded as having developed collaborative relations with each

---

6 Because only approximately 2% of developers make their technical skill profiles publicly accessible at SourceForge.net, we inferred their technical skills from their past project experience. The specific variables that we constructed for hypothesis testing are described in more detail in subsequent sections.
other.\textsuperscript{7} Given the self-organizing nature of OSSD project teams, developers may join a project at their own discretion. However, when this joining decision is made, the set of project members may include the original initiator of the project as well as other developers who have joined the project before the focal developer makes his joining decision. In other words, he may join a project due to his sensitivity toward cohesion with the initiator or with the other developers who have already joined. Given the differences in roles that initiators (i.e., leadership or administrator roles) and other members play (i.e., co-worker roles), we distinguish between ties with the project initiator and with other developers that have subsequently joined the project prior to the focal developer joining the project. In other words, the focal developer may have a collaborative tie with the initiator and/or with the non-initiator members of the project.

For each pair of project member and focal developer we collected seven measures of past joint project activity that tapped into either the outcome of the prior collaboration or the strength of the collaboration tie. In order to operationalize collaborative tie outcome, we characterized the nature of the past collaborations based on measures of OSS project success (Crowston et al. 2006). The extent to which past collaborative experiences had been successful was measured by the amount of code released ($\text{CodeBytes}$), the number of software downloads by users ($\text{Downloads}$), whether or not the project resulted in a successful release of the working program code ($\text{HasRelease}$), and the development status of the project ($\text{DevelopmentStatus}$). We measured strength of collaborative tie as a combination of the frequency of collaborative interactions and the closeness of the collaborative relationship (Cummings and Kiesler 2007, Granovetter 1973, Hansen 1999). The intimacy and amount of time spent in cultivating the tie was measured through the number and duration of the projects they collaborated on ($\text{NumCollaborations}$ and $\text{Duration}$). Because we expect project administrators to interact more frequently and develop closer working relationships in order to coordinate the project we also tracked whether the focal developer and project member jointly had project administration responsibilities in prior projects.

\textsuperscript{7} This operationalization of past collaborative ties does not take into account whether the two developers have actually worked collaboratively during their concurrent membership in a project. Consequently, this operationalization would underestimate the impact of prior collaborative ties, rendering the test of hypotheses...
(BothAdminRole). Since a developer-initiator or developer-developer pair could have collaborated on multiple projects, we use the average of all measures except NumCollaborations.

We conducted factor analysis on these measures for all pairs of developers who have collaborated with each other at SourceForge.net to determine whether the joint project activity measures could be grouped into the higher order factors of collaborative tie strength and collaborative tie outcome as expected. Two factors emerged that collectively explained almost 70% of the variance (see Table 1). The first factor, TieOutcome, measures whether or not the outcome of the past collaboration between the focal developer and the project member was positive, that is whether the project was successful as measured through number of downloads, among other factors. The second factor, TieStrength, measures the closeness of the work relationship (BothAdminRole) and the frequency of interaction (NumCollaborations). The resultant TieStrength measure is similar to measures of tie strength adopted in studies of the impact of tie strength within organizations and reflects the frequency (NumCollaborations) and intensity (BothAdminRoles) of the joint collaboration (Cummings and Kiesler 2007, Hansen 1999).

Contrary to what we expected, the duration of the joint projects loaded onto TieOutcome rather than TieStrength. We suspect that the actual project duration may not be an accurate indicator of the actual interaction frequency or intensity given the discretionary nature of participation in OSSD projects. Rather, since failed projects may be abandoned earlier, longer project durations may be indicative of positive project outcome. Hence, we include it as an item measure of TieOutcome.

The composite measures for TieOutcome and TieStrength were constructed using factor scores based on the factor loadings shown in Table 1. As discussed above, we distinguished between prior ties with the project initiator and those with other non-initiator members of the project team by prefixing the variables with Initiator or Other. Also, since there could be multiple non-initiator members that have already joined the project team, we aggregate the measures by summing the values for all developer-member conservative.
pairs. Collaborative tie strength and outcome can be positive or negative depending on the nature of past collaborations and was set to zero when the focal developer and the existing project member had not jointly participated in any common projects.

Status Cues. Project members’ status cues result from their prior project collaboration experiences within the SourceForge.net OSSD network. We constructed a measure of project members’ ties based on the number of distinct developers in the OSSD network with whom each member has had previous joint project collaboration experience at SourceForge.net (TieAmount). Again we distinguish between ties of the initiator and those of the non-initiator members and prefix the variable with Initiator or Other. Also, since there could be multiple non-initiator members in the project team, we use the sum of ties across all members.

In addition, although our focus is on status cues resulting from developers’ prior collaborative ties, we also include measures of developers’ experience since these may influence others’ perception of their status within the OSSD community. We measured the prior collaboration experiences of the project initiator and non-initiator members using the active duration since their first actual participation in an OSSD project (other than the focal project) at SourceForge.net (InitiatorExpActiveT and OtherExpActiveT) as well as the number of projects on SourceForge.net in which they have participated.

---

8 We used other aggregate measures such as maximum (for best experience), minimum (for worst experience) and average (for average experience) to compute the tie measures between developers and non-initiator members. Sensitivity analyses suggested that the results reported are robust to different operationalizations.

9 We also tried including developers’ indirect ties (i.e., ties with other developers through a common collaborator). However, the correlation between the direct tie amount and the indirect tie amount was very high ($\rho = 0.85$ for initiators and $\rho = 0.85$ for other pre-existing members). Consequently, our analyses are based on the operationalization that only accounts for the number of direct ties. Sensitivity analyses suggest that the results are robust to changes in this operationalization.
in the past \((InitiatorExpP \text{ and } OtherExpP)\). Again, in cases where there are multiple non-initiator members in the project team, we use the sum of ties across all members.

**Control Variables**

While our study focuses on examining the impact of cohesion and status cues shaped by prior collaboration experiences of developers within the OSSD network, developers may also rely on other attributes in evaluating which project to join. Hence, we controlled for three categories of factors that may influence this choice – factors related to the project in question, those related to the focal developer, and those that relate to the fit between project and focal developer.

*Project-Related Factors*

**Code Availability.** The presence of software code may reduce the uncertainty regarding the nature of the project as well as the ability of the project members to translate project goals into actual working code. The source code also gives developers the opportunity to obtain additional technical information regarding the project itself. Some projects may be registered on SourceForge.net only after some working code has been independently developed by the initiator. In other cases, a project may be a fork of an existing project hosted at SourceForge.net or other sites. In both cases, projects would be more likely to have code present either before or within a short period of project initiation on SourceForge.net. Hence, we used a binary variable \(PreexistingCode\) to control for the initial presence of a project’s source code. \(PreexistingCode\) indicates whether the project’s first release date was prior to its registration date. We also controlled for whether the project has released software code \(\text{CodeReleased}\) after the project was registered at SourceForge.net by the focal date, that is, the date on which the focal developer joins the project.

**Project Activity.** We also controlled for other project attributes that would affect the visibility of the project to potential developers. For example, a factor that may influence project visibility on

---

10 Although the project founding experience of the initiator may also be an indicator of status and influence whether developers join the project, only one third of the projects hosted at SourceForge.net have their initiator information recorded in the database. Thus, due to data unavailability we were unable to consider the initiator’s experience in
SourceForge.net is the information related to project ranking. SourceForge.net calculates project scores based on traffic, communication and development statistics. Each project’s score is compared to those of other projects for the same time period to determine its ranking. Projects with high activity ranks are included in SourceForge.net’s top project listing, which makes them more visible to potential developers. To capture this, we use a categorical variable (ActivityPercentile) that distinguishes between low, medium and high rankings.\footnote{Each project’s ranking percentile information is displayed on its summary page. We also tried the nominal scale for ActivityPercentile and the results are unchanged.}

**Project Popularity.** Developers may consider indicators of project popularity when deciding which projects are likely to result in positive outcomes. For example, developers may rely on the expressed preferences of other developers to reduce the uncertainty regarding viable project choices and perceive more favorably those projects where others have already joined. Therefore, we also control for TeamSize, operationalized as the number of developers in the project on the focal date. Project popularity may also be partially determined by the software domain. Some software domains may be more popular than others, increasing their attractiveness to potential developers. We controlled for domain popularity using the ratio DomainPopularity, which was computed as the proportion of developers for the top 1000 projects who work on the focal project’s domain. For example, if a total of 2000 developers participate in the top 1000 projects at SourceForge.net and among the 2000 developers 500 are members of the projects in the “Internet” domain whereas 200 are members of the “security” domain, DomainPopularity would be 0.4 for the Internet domain (i.e., 500 / 2000 = 0.4) and 0.1 for the security domain (i.e., 200 / 2000 = 0.1).

**Project Information Availability.** We included project attributes affecting the amount of information that is available to developers. We measured the level of details available in the project description that would facilitate information gathering required for making a joining decision using DescDetail, which is operationalized as the number of characters in the project description.
Other Controls. We measured other factors that may influence developer decisions to join a particular project. We controlled for whether a project accepts donations from users using a binary variable AcceptDonation. The availability of donations as a source of external resources may influence a project’s attractiveness to potential developers. Finally, ProjectAge represents the number of days between the project’s registration and the focal date. Some developers may prefer joining a relatively new project whereas others may prefer waiting for a period of time to observe the progress of the project and reduce any uncertainty before making their joining decisions.

Developer-Related Factors

Besides the project attributes as discussed above, we also controlled for additional factors related to the focal developer that may affect his decision to join a particular project. A developer who has been actively participating in projects and has more experience in OSSD may consider new projects differently compared to less experienced developers. Therefore, we measured the experience of the developer in terms of how long he has participated in OSSD at SourceForge.net. DeveloperExpActiveT is the number of months since he first became a member of an OSSD project on SourceForge.net.\textsuperscript{12}

Project-Developer-Related Factors

We considered the fit between developers’ skills and the focal project’s requirements in order to control for the impact of developer expertise and interest on project selection. These were captured in four variables that reflected how well the technical details in terms of domain (DomainMatch), programming language (PLMatch), intended audience (AudienceMatch), and application platform (OSMatch) of developers’ past projects matched those of the focal project.

\textsuperscript{12} Although we measured initiators and other project members’ experience using both the number of prior projects as well as the time engaged in project participation, we only use time-based experience to control for developer experience. This is because, as mentioned earlier, when constructing the sample we sampled only the developers with past project experience in order to be able to compute the matching variables based on the properties of their past projects. However, since the event sample consisted of not only experienced but also inexperienced developers, controlling for developers’ project experience in the analysis would lead to a large but biased estimate due to the fact that their project experience was already controlled for during the sampling process. Therefore, we did not include the focal developer’s experience in terms of projects in the analysis.
Because using binary measures to represent whether at least one past project that the developer participated in matches the technical detail of the focal project may introduce a bias in the likelihood of fit toward more experienced developers, we computed the matching variables in such a way as to be less sensitive to developers’ experience. The DomainMatch scores were calculated as the percentage of a developer’s past projects that have at least one domain in common with the focal project. If the focal project is related to multiple areas (e.g. both Databases and Internet), we computed the dominant match for each of these areas and selected the maximum so as to capture the dominant matching concept. For instance, assume the focal project’s domains include D1, D2 and D3, and the developer’s past projects include P1 and P2. Further assume P1 covers domains D1 and D4, and P2 covers domains D1 and D2. The matching score for D1 would be 1 because all of the developer’s past projects are related to D1; the score for D2 would be 0.5 because only one of the two prior projects covers D2; the score for D3 would be 0 since none of the developer’s past projects is related to D3. Thus, DomainMatch for the project developer pair would be 1, the maximum of the scores for D1 (1), D2 (0.5) and D3 (0). The same approach was applied to compute the match on other technical details. These scores are continuous values ranging from 0 to 1 with higher values indicating higher degree of match.13

In addition, because domains and operating systems are defined at several levels on SourceForge.net, we computed the matching variables for these two types of technical details at both the top level and the second level, which are highly correlated (Domain: $\rho = 0.89$, $p < 0.001$; OS: $\rho = 0.64$, $p < 0.001$). Hence, we only kept the top level domain matching and OS matching variables in subsequent analyses. The summary of the measures are shown in Table 2.

13 A developer’s interests may change over time, in which case considering all of his past projects would not faithfully represent his true interests. In such cases, constructing the match variables with a developer’s most recent project may be more accurate. We tried this alternative operationalization – using only the developer’s most recent project to calculate the match between the requirements of the focal project and the developer’s interests. However, the correlation between the dominant and recent match scores was very high ($\rho = 0.90$) suggesting that, at least for developers in our dataset, developers’ interests were quite stable over time. Sensitivity analyses suggest that the results are robust to changes in this operationalization.
3.3. Analytical Procedures

Given that the dependent variable is a binary variable representing the joining event, we conducted logistic regression analysis to test our hypotheses. However, the conventional logistic regression approach with random sampling is inappropriate here due to the rarity of the joining event, which causes logistic regression to underestimate event probabilities (King and Zeng 2001). One reason for the unreliable estimates under random sampling is that the maximum likelihood estimators obtained by logistic regression are biased not only in samples with fewer than 200 observations but also in large samples where the proportion of positive outcomes in the samples is very small. Random sampling, in which the selection rule is independent of all other variables tends to generate too few instances of the event in the sample to make logistic regression analysis an optimal approach. For instance, with approximately 2,300 sample projects, 170,000 sample developers and 100 days, there would be over 39 billion (i.e., $2,300 \times 170,000 \times 100$) date-project-developer triads in total, of which the joining event occurs only in approximately 1,200 instances (i.e., less than $3 \times 10^{-6}$% of cases).

A data sampling strategy to overcome these problems is choice-based (or endogenous stratified) sampling (King and Zeng 2001, Manski and Lerman 1977). Unlike random sampling, choice-based sampling strategically selects observations based on values of the response variable $Y$. Compared with observations where the response variable is 0, observations with a response variable of 1 carry much more information for the estimation of the variables influencing the occurrence of the event. The strategy is to construct a sample by collecting a fraction $e$ of the observations with $Y = 1$ (the event sample) and a fraction $c$ of the observations with $Y = 0$ (the non-event controls), such that $e$ is much larger than $c$.

Our dataset includes 1178 observations representing events where a developer joined a project on a particular date (i.e., $Join = 1$). We then matched each event triad with five control triads whose date is the same as the focal date in the event triad. We also ensured that control triads have similar (or dissimilar) characteristics as the event triads. In particular, for one control triad we ensured that the
existence (or nonexistence) of prior collaborative ties between the developer and the initiator was the same as that in the event triad. In another control triad the existence (or nonexistence) of prior collaborative ties between the developer and the non-initiator members was the same as that in the event triad. In addition, three random triads are selected for each event triad. The choice-based sampling procedure produced a sample of approximately 7,000 triads.14

Corresponding to the choice-based sampling technique, we adopted the weighted exogenous sampling maximum-likelihood (WESML) estimator (Manski and Lerman 1977) as a validated approach adopted in prior literature (e.g., Singh 2005). The WESML estimate maximizes the weighted pseudo-likelihood function that weighs each observation in the sample with the number of population observations that it represents.15 For example, the weight of a sample triad that represents 100 potential triads in the entire population is assigned 10 times the weight of a sample triad that represents 10 population triads. In addition, because in the choice-based sampling process a project may be sampled multiple times, we used the generalized estimating equations (GEE) method (Liang and Zeger 1986) to calculate the standard errors without assuming independent errors among observations.

In order to ensure the robustness of the estimation results with respect to the choice-based sampling procedure, we drew 1000 bootstrap choice-based samples of approximately 7000 observations to derive the bootstrap mean and the confidence intervals for each parameter estimate (Efron 1982, Efron and Tibshirani 1986). Statistical significance of the parameter estimates is based on the bootstrap confidence intervals.

4. RESULTS

4.1. Data Sample and Descriptive Statistics

Table 3 presents the distribution of software domains of the projects in our sample. The sample projects covered all 19 top-level domain categories, among which Software Development, Internet, and

---

14 Since we have 5 control triads for each event triad, the choice-based sample size is 1178 × 6 = 7068. However, due to possibility of missing controls (i.e., when an observation that meets the control sampling criteria does not exist), a few observations may be dropped.
System had the highest frequencies. Table 4 summarizes the descriptive statistics obtained from the 1000 bootstrap samples. A total of 520 projects had attracted additional developers during the data collection period and on average about 2 additional developers joined each of these projects.¹⁶

Table 5 presents the bootstrapped means of the pairwise correlations between variables. Most variables are weakly correlated with correlations below 0.5. The highest correlation among the independent variables is between OtherTieAmount and OtherExpP (ρ = 0.77, p < 0.001). There also seems to be moderate correlation between OtherExpP and OtherExpActiveT (ρ = 0.69, p < 0.001) as well as between InitiatorExpP and InitiatorExpActiveT (ρ = 0.67, p < 0.001). We conducted diagnostic checks for multicollinearity by obtaining the bootstrap means of the tolerances (Greene 2000) and the variance inflation factors (Greene 2000, Mansfield and Helms 1982) of all predictor variables. The resulting tolerances were all above 0.20 and the highest variance inflation factor value was 3.60, which is below the commonly used cut-off value of 10 (Neter et al. 1996), suggesting that the potential bias in parameter estimates due to multicollinearity is not problematic.

4.2. Tests of Hypotheses Predicting Developer Joining Events

As discussed in previous sections, we distinguished between the initiator and non-initiator members who have already joined the project in testing the hypotheses. To test developer sensitivity to cohesion cues, that is their reliance on direct prior collaboration experiences with current project members when joining new projects, we investigate the coefficients of the parameters related to collaborative tie

---

¹⁵ For more technical details on WESML refer to King and Zeng (2001).
¹⁶ Some projects may be founded without any intentions to continue development or attract developers to the team (e.g., class projects). We conducted a robustness check by including only the 520 projects that had successfully attracted additional developers. The major findings presented in the paper are largely consistent with slight differences in the level of statistical significance of some parameter estimates.
strength (H1) \((InitiatorTieStrength, \text{ and } OtherTieStrength)\) and the actual collaboration outcome (H2) \((InitiatorTieOutcome, \text{ and } OtherTieOutcome)\) of developers’ past collaborative experiences. To test developer sensitivity to status cues, that is their preference for collaborating with developers who have demonstrated success in past collaborations (H3), we investigate the impact of the amount of ties to other developers forged through previous collaborations \((InitiatorTieAmount, \text{ and } OtherTieAmount)\) as well as the previous experience of the project members on other SourceForge.net projects \((InitiatorExpActiveT, \text{ OtherExpActiveT}, InitiatorExpP, \text{ and } OtherExpP)\). The results of the logistic regression are presented in Table 6.

\[
InitiatorTieStrength \text{ is positive and significant } (\bar{\beta} = 0.708, p < 0.05) \text{ indicating that the strength of a collaborative tie between the project initiator and the developer has a positive influence on the developer’s decision to join the project; whereas OtherTieStrength does not significantly influence the developer’s joining behavior } (\bar{\beta} = 0.367, \text{ ns}). \text{ In short, developers are more likely to join a project initiated by someone with whom they have developed strong ties through repeated collaborations and joint administration of past projects. Developers however are not influenced by the strength of ties with non-initiator developers already in the project. In summary, H1 is supported: when a developer has a strong tie with the project initiator based on joint project administration and frequent collaborative experiences in the past he is more likely to join the project. H2 however was not supported. Neither InitiatorTieOutcome } (\bar{\beta} = -0.027, \text{ ns}) \text{ nor OtherTieOutcome } (\bar{\beta} = 0.985, \text{ ns}) \text{ affects the likelihood that a developer joins the new project. Developers were not influenced by whether or not their past collaboration projects with either the initiator or other members of the new project were successful in terms of the project outcomes.}
\]

H3 posited that project member status based on past collaborative experiences measured as the number of collaborative ties to other developers in the SourceForge.net developer network and the extent
of project-related experience (in terms of time and number of projects) would positively affect a developer’s likelihood of joining a project. There was no significant effect of the initiator’s status operationalized as the number of collaborative ties on the likelihood that a developer would join the project \(\text{InitiatorTieAmount}: \beta = -0.19, \text{ns}\). Furthermore, the initiator’s status operationalized based on past project experience did not have the expected significant impact \(\text{InitiatorExpActiveT}: \beta = -0.014, \text{ns}\;\text{;}\;\text{InitiatorExpP}: \beta = -0.009, \text{ns}\). Contrary to what we predicted, the non-initiators’ status operationalized through the number of collaborative ties has a significant but negative impact on developer joining \(\text{OtherTieAmount}: \beta = -0.384, p < 0.05\). A unit increase in \text{OtherTieAmount} decreases the likelihood of developer joining by 31.6%. Non-initiators’ status operationalized as past project experience has a significant positive impact on developers’ joining behavior \(\text{OtherExpActiveT}: \beta = 0.027, p < 0.01\;\text{;}\;\text{OtherExpP}: \beta = 0.678, p < 0.05\). Thus, H3 is partially supported when status of non-initiator members is operationalized based on the number of past project collaborations and the actual time spent in these endeavors, but not when operationalized as the number of developers they have developed ties with through these projects.

We further examined the impact of additional project and developer attributes on the likelihood that a developer joins a project in order to arrive at a deeper understanding of how they influence developer choice of new OSSD project teams. We focus on factors that were found to have a significant impact among those attributes that we considered. We examined visible project attributes that may signal a project’s potential to result in positive collaborative process and outcome – namely that related to code availability and project popularity. The significant and positive parameter estimate for \text{CodeReleased} \(\beta = 1.280, p < 0.05\) suggests developers are more likely to join a project that has released some software code. The significant positive effect of \text{TeamSize} \(\beta = 2.550, p < 0.01\) indicates that a developer prefers joining those newly-initiated projects that have already been successful at attracting additional developers. A unit increase in project team size increases the likelihood of developer joining
by almost 12 times. Other project attributes related to the activity levels and available resources of a project were also found to affect the likelihood of a developer joining. Developers were more likely to join projects earlier ($ProjectAge: \beta = -1.782, p < 0.01$) and active projects ($ActivityPercentileLow: \beta = -0.974, p < 0.01$). Furthermore, a project set up to accept donations from users is more likely to attract additional developers into the team ($AcceptDonation: \beta = 0.655, p < 0.05$).

We also examined how developers’ prior experiences may influence their decisions to join new OSSD project teams. $DeveloperExpActiveT$ is negatively associated with developers’ joining behavior ($\beta = -0.042, p < 0.01$), suggesting that less experienced developers are more likely to join a new project. The negative estimates for $AudienceMatch$ ($\beta = -1.077, p < 0.01$) and $OSMatch$ ($\beta = -0.774, p < 0.05$) indicate that developers tend to participate in projects designed for intended audience and operating systems that are different from that of their previous projects.

In summary, the results suggest that OSS developers prefer joining a project whose initiator has developed a strong tie with them through repeated collaborations and shared administration of past projects. They are also more likely to join a project whose non-initiators have more project experience. In addition to ties with the initiator and experience of the non-initiators, other project characteristics such as team size, code release, project age, and whether projects accept donations also seem to play a role in influencing developer decisions to join an OSSD project.

5. DISCUSSION

In this study, we examined the impact of the online collaboration network on how individuals embedded in the network self-organize into newly-formed teams in the OSSD context. Specifically, we empirically investigated whether past ties with and status of existing project members influence developers’ decisions to join the project. Overall, our results supported our main thesis that the network of collaborative ties affects developer decisions to join new projects. We found that developers rely on both their past collaboration experiences with project initiators and other members’ status cues when
choosing which project to join. We conclude by offering some general observations regarding developer choice of new OSSD projects based on the collective results.

In OSSD projects, the most critical resource is its members. We find that developers rely on a variety of cues when evaluating the attractiveness of a project based on its potential collaborators, and that the cues they focus on may be different depending on the role of the potential collaborator (i.e., project leader/initiator vs. other developer). In the case of the role of project leader, developers are prone to participate in new OSSD projects that are initiated by developers they have interacted intensely with in the past regardless of how successful the previous project has been (H1, H2). A developer is more likely to join a project when he has developed a strong collaborative tie with the project initiator through repeated collaborations and shared administrative roles in the past, demonstrating the vital role of initiators in attracting new developers. There may be various reasons why developers repeat collaborations with project initiators they have worked with in the past. First, software development is not only a production process but also a social process that relies heavily on interpersonal communication and coordination (Curtis et al. 1988, Robey and Newman 1996, Sawyer et al. 1997, Sawyer and Guinan 1998). Moreover, in the OSSD context, the difficulty inherent in the social process becomes even greater than in traditional software projects since members of an OSSD project are typically from geographically dispersed locations (Crowston and Scozzi 2002) and have limited (if any) face-to-face interactions (Scacchi 2002). From the developer’s perspective, a large amount of uncertainty exists with regard to how smooth and how efficient it will be to interact with the existing members of a project. Developers who have built strong ties with the project initiator through past interactions and joint project administration experiences may have developed a sense of trust and norms of behavior with the project initiator that can improve team effectiveness (Adler and Kwon 2002). Their familiarity with and trust in the project leader can mitigate the risks and reduce the uncertainties in the collaboration process. Second, the positive impact of strong ties with the initiator may also be caused by a sense of belonging to the project group that developers participated in with the initiator. A survey of Linux developers found that active developers identified strongly not only with the Linux community broadly but also with the
specific subsystem team (Hertel et al. 2003). This sense of group identity will lead OSS developers to evaluate the projects initiated by those they perceive to be in-group members more favorably than projects initiated by out-group members (Hogg and Abrams 1988). However, whether or not the past collaborations that developers engaged in with the project initiator were successful in outcome did not seem to influence developers’ joining behaviors (H2). If developers were motivated mainly by the use-value need in participating in OSSD, i.e., successfully developing software that addresses their personal needs, then we would expect that the positive outcome associated with prior collaborative ties with the initiator would impact project joining decisions. The fact that this relationship is insignificant may indicate that for OSSD participants, the eventual outcome of the OSSD project might be less important than the fun, enjoyment and learning benefits they derive through the process of jointly developing code and participating in OSSD projects (Lakhani and Wolf 2005). An alternative explanation for the apparent lack of impact of prior collaboration success may be that our operational measures do not accurately reflect actual project success and the outcome dimensions that OSS developers value. Because most OSS projects release code early and often, our outcome measure, which places heavy emphasis on code released, may not be an indicator of success, but rather of ongoing development processes. Developers were not influenced by status cues of project initiators when selecting projects to join (H3). In other words, developers relied on their own personal experiences and joined projects initiated by developers they knew and had built strong collaborative ties with, but did not rely on the implicit recommendation of others in the OSSD network based on the initiator’s prior project collaboration experiences.

A different pattern emerges in the case of developers’ choice of non-initiator collaborators in new OSSD projects. Collaborative tie strength with other members of the project did not have a similar effect on whether a developer joined a new project (H1). This may be because leaders play a more critical role in OSS development process relative to other developers. The leader, who in most cases is also the project initiator, guides the overall direction of the project and influences how the project will progress in the future (Lerner and Tirole 2002, Nakakoji et al. 2002, Xu et al. 2005). Hence, developers may be less sensitive to whether or not they share a strong collaborative tie with other members of a project because
they perceive their role in project communication and coordination to be less critical than that of the project initiator. In addition, initiators tend to be more committed to the project than other members of the project who may lose their interest in the project after joining it. Whether or not a prior collaboration with non-initiator members of a project resulted in a positive outcome also did not seem to affect developers’ joining decisions (H2). While prior collaborations may have driven the choice of project initiator, given the scale of the OSSD collaboration network where the focal developer does not know the majority of others, relying on prior collaboration experiences in evaluating the non-initiator members of a new OSSD project may be intractable. Faced with such uncertainty, a developer may rely on status cues of the non-initiator members when deciding whether or not to join a particular project (H3). In particular, developers are more likely to join a project whose non-initiator members have more project experience, based on both actual time invested in project participation and the number of projects. Developers may perceive non-initiator members with accumulated OSSD project experience to be more competent and capable, and also rely on their experience to judge the attractiveness of the project and its likelihood of success. However, developers were less likely to join when non-initiator members had a greater number of past collaborative ties. One possible explanation for this may be related to our operationalization of the status cues. While all three indicators of status are based on involvement in past projects, the number of past collaborative ties may also dilute the perceived experience level of the project members. In other words, insofar as the competence and skills of the non-initiator members based on their previous project experiences have been captured through the measures of their actual project experience, the number of past collaborative ties may be measuring the significance of the contributions made to the projects they have participated in. More collaborative ties could be construed as an indicator of having played a less significant role in the previous collaborations, and hence developers may perceive non-initiator members with greater number of ties given the same project experience as less competent and capable.

In summary, the results indicate that developers may use different cues in evaluating potential collaborators in new OSSD projects, depending on their perceived role within the projects. Developers
join projects that are initiated by developers they have established strong collaborative ties with, but rely on other status cues when evaluating other members of a project.

We offered one possible explanation for the significant negative impact of the number of collaborative ties that non-initiator members of a project have on the likelihood of joining that project above. An alternative explanation is offered based on expectancy value models (Karau and Williams 2000). While from the project perspective a greater number of collaborative ties may increase the likelihood of technical and commercial success (Grewal et al. 2006), from the perspective of a potential developer contributing to the project, the presence of other skilled developers with greater access to resources might reduce his opportunities to contribute to and gain recognition from the project. According to expectancy value models the motivation to contribute to a group will be positively influenced by the developers’ expectation that their contributions are unique and hence valued (Karau and Williams 2000). This explanation appears somewhat plausible when considered together with the significant negative effect of project age (ProjectAge) on the likelihood that a developer joins the project. Developers were less likely to join projects when projects were already well under way. This may be because the earlier they join, the more impact their contributions may have and hence the more visible their contributions to the project may become as the project progresses and keeps attracting more developers in the future (Lerner and Tirole 2002). Newer projects may also have lower technical barriers to joining as the software code and architecture are relatively easier to understand when they are still in their early stages. As the software grows more complex, fewer developers are able to fully understand the architecture and effectively contribute code because of a high learning curve (Kohanski 1998, von Krogh et al. 2003). Hence, developers may have lower expectations of being able to make meaningful contributions that have an impact as the project age increases and thus be less likely to join the project.

Developers also are more likely to join projects they are able to judge as having a higher probability of success. We found that a project with more developers is more likely to attract additional developers than a project with fewer developers, a finding consistent with the “rich get richer” mechanism
proposed by Barabási and Albert (1999) for many self-organizing networks.\textsuperscript{17} Developers are more likely to join a project that exhibits higher than average development activity levels and has released some initial software code that may outline the functionalities envisioned by the project initiator and demonstrates the potential merits of the project (Lerner and Tirole 2002). This is consistent with the argument that some minimal code needs to be assembled in order for the project to receive reaction from the OSSD community (Raymond 2001). The existing activity and initial code may facilitate developers’ judgment regarding a project’s probability of success. Additionally, projects accepting donations from users are more likely to attract developers to join as such donations may improve a project’s chances by providing needed external resources.

Finally, developers also seek to increase the variety of their OSSD experience and acquire new skills. They are less likely to join new projects if these are developing software that is based on an operating system and intended for an audience for which they have previously developed software as members of a different project. However, with increased experience developers may no longer need to participate in new projects for the purpose of learning. Developers with longer participation history are less likely to join new projects, as they may have gained some expertise and experience in OSSD and may consequently prefer to continue with their current projects or initiate their own projects rather than joining others’ new projects.

The study has both theoretical and practical implications. Theoretically, this study contributes to the growing stream of studies examining why and how the structure of social relations impact individual behavior in a self-organizing network by employing a social network perspective to examine the interactions among interdependent OSS developers embedded in an online collaborative network. The

\textsuperscript{17} On the surface this may appear to be inconsistent with the argument that developers wish to increase the chance that their contributions will be meaningful and visible and hence are less likely to join projects where the non-initiator members have more collaborative ties within the OSSD network. However, having more developers on the team who have less experience and resources would not necessarily result in lower chances for one to make meaningful contributions. In short, it is not the number of developers per se but the potential for their differential ability to access resources in the OSSD network, based on their collaborative ties, that determines whether one’s chance for making an impact on the project outcome is in fact reduced.
findings from our study also contribute to an enriched understanding of the ecology of OSS projects by providing one perspective on attributes of new projects and their members that attract developers. Although a growing body of research has explored the motivations of OSS developers, little research to date has attempted to understand how developers choose which of the myriad of possible projects available to contribute to. Additionally, our paper also provides empirical evidence for one mechanism that could explain the reason behind the preferential attachment processes leading to the formation of scale-free networks prevalent in digital and social networks. OSS developers choose to continue collaborating with project leaders with whom they have developed strong collaborative ties. This finding is consistent with research in the formation of founding teams for new firms that finds a tendency for entrepreneurs to prefer people with whom they already have strong ties (Ruef et al. 2003).

Although our main focus here was to investigate developers’ decisions to join newly-created projects and the results may not apply to developers joining established projects, our findings also have important practical implications that may help new projects obtain early momentum. Developers interested in launching new OSS projects may be well-advised to first establish strong collaborative ties with other developers in the OSSD network, and thereby increase the size of the development team through these collaborative ties. In addition, releasing some early working code of the project would increase the probability that the project would attract additional developers.

We end our discussion by noting some limitations and additional directions for future research. First, in this study we focused on the factors that affect a developer’s choice of new projects to join and treated the new project team’s existing members and initiator as passive entities. The significant impact of collaborative tie strength is also consistent with project initiators seeking out developers with whom they have worked in the past to help them in their new project. While this is plausible, we found in a survey of our sample project administrators that this rarely occurred, and hence the one-sided process presented in this study was more appropriate in our context of archival field study design. Future research employing other methods should extend the model of new project joining to account for the active role that existing project members and the initiator play in new developers joining a project. A second
limitation is that we only examined joining behavior within the initial months after project initiation. The joining behavior may differ depending on stages of project development. While controlling for development stage would offer richer theoretical insights, practically, many newly registered projects do not define their development stages explicitly, which limits our ability to incorporate this factor into the analysis. An important extension of this paper would be to study the effect of developer joining behavior on the network structural characteristics within project team as well as in the OSS developer network.

Third, our data only include information available from SourceForge.net. Even though SourceForge.net is currently the largest repository of open source software, constraining the data collection to one site limits our ability to capture the interactions among developers at other OSS hosting sites. For example, even if two developers have no prior collaborations at SourceForge.net, they may have already formed relations with each other through collaborations on other projects hosted elsewhere. But since we cannot observe their past collaborations and code them as having no ties, our estimates of the impact of past collaborative ties would be underestimated and the test of hypothesis conservative. Finally, an interesting avenue of future research would be to investigate how early team formation influences a project’s subsequent performance and sustainability.

Overall, this research contributes to a richer understanding of the assembly mechanisms for self-organizing teams embedded in an online collaborative network. It is one of the few studies that examined how social networks influence the joining activity of OSS developers. The findings in the research also offer some guidelines to OSSD project managers that can help them attract developers into the teams and gain early momentum.
REFERENCES


Stewart, K. J., A. P. Ammeter and L. M. Maruping. 2006. Impacts of license choice and organizational sponsorship on user interest and development activity in open source software projects. *Information Systems Research* 17(2) 126-144.


Table 1. Composite Measures of Strength and Outcome of Prior Collaborative Ties with Project Members

<table>
<thead>
<tr>
<th>Variable</th>
<th>Factor 1 Tie\text{Outcome}</th>
<th>Factor 2 Tie\text{Strength}</th>
</tr>
</thead>
<tbody>
<tr>
<td>Downloads</td>
<td>0.958</td>
<td>0.002</td>
</tr>
<tr>
<td>Duration</td>
<td>0.647</td>
<td>0.036</td>
</tr>
<tr>
<td>HasRelease</td>
<td>0.893</td>
<td>-0.078</td>
</tr>
<tr>
<td>DevelopmentStatus</td>
<td>0.719</td>
<td>-0.264</td>
</tr>
<tr>
<td>CodeBytes</td>
<td>0.968</td>
<td>0.034</td>
</tr>
<tr>
<td>BothAdminRole</td>
<td>-0.162</td>
<td>0.695</td>
</tr>
<tr>
<td>NumCollaborations</td>
<td>0.110</td>
<td>0.826</td>
</tr>
<tr>
<td>Eigenvalues</td>
<td>3.648</td>
<td>1.223</td>
</tr>
<tr>
<td>% Variance Explained</td>
<td>52.11 %</td>
<td>17.47 %</td>
</tr>
<tr>
<td>Cum. Variance Explained</td>
<td>52.11 %</td>
<td>69.58 %</td>
</tr>
</tbody>
</table>
Table 2. Summary of Measures

<table>
<thead>
<tr>
<th>Variable</th>
<th>Operational Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Join (DV)</strong></td>
<td>Binary variable, which equals 1 if the developer joined the focal project on a specific date (focal date below) (between September 30, 2005 and January 5, 2006), 0 otherwise.</td>
</tr>
<tr>
<td>**Cohesion Cues based on Direct Past Ties between Developer and Project Members (H1, H2)**a</td>
<td></td>
</tr>
<tr>
<td><strong>InitiatorTieOutcome</strong></td>
<td>The outcome of a collaborative tie with the project initiator that is associated with the success level of the project. See Table 1.</td>
</tr>
<tr>
<td><strong>InitiatorTieStrength</strong></td>
<td>The strength of a collaborative tie with the project initiator that is related to the frequency and closeness of past collaboration. See Table 1.</td>
</tr>
<tr>
<td><strong>OtherTieOutcome</strong></td>
<td>The sum of the outcome of collaborative ties with the non-initiator members associated with the success level of the project. See Table 1.</td>
</tr>
<tr>
<td><strong>OtherTieStrength</strong></td>
<td>The sum of the strength of collaborative ties with the non-initiator members related to the frequency and closeness of past collaboration. See Table 1.</td>
</tr>
<tr>
<td><strong>Status Cues based on Project Members’ Prior Collaboration Experiences (H3)</strong></td>
<td></td>
</tr>
<tr>
<td><strong>InitiatorTieAmount</strong></td>
<td>The natural log of the number of distinct developers (+1) who have collaborated with the project initiator on OSS projects at SourceForge.net.</td>
</tr>
<tr>
<td><strong>OtherTieAmount</strong></td>
<td>The natural log of the number of distinct developers (+1) who have collaborated with the non-initiators on OSS projects at SourceForge.net.</td>
</tr>
<tr>
<td><strong>InitiatorExpActiveT</strong></td>
<td>The number of months since the project initiator first became involved in a project as a member at SourceForge.net.</td>
</tr>
<tr>
<td><strong>OtherExpActiveT</strong></td>
<td>The average of the non-initiators’ experience in terms of the number of months since they first became involved as members of a project at SourceForge.net.</td>
</tr>
<tr>
<td><strong>InitiatorExpP</strong></td>
<td>The natural log of the number of prior projects (+1) that the project initiator has participated in at SourceForge.net.</td>
</tr>
<tr>
<td><strong>OtherExpP</strong></td>
<td>The average of non-initiators’ experience in terms of the natural log of the average number of prior projects (+1) that they have participated in on SourceForge.net.</td>
</tr>
<tr>
<td><strong>Control Variables</strong></td>
<td></td>
</tr>
<tr>
<td><strong>DomainPopularity</strong>b</td>
<td>Popularity of the project’s domain as measured by proportion of developers working on the top 1000 projects within the domain category. DomainPopularity is set to 0 if the project’s domain is undefined.</td>
</tr>
<tr>
<td><strong>ActivityPercentile</strong>b</td>
<td>A categorical variable that is based on percentage of projects hosted on SourceForge.net that have lower project scores than the focal project – “low” for projects with activity scores below the 50th percentile, “medium” for those between the 50th and 90th percentiles, and “high” for those above the 90th percentile.</td>
</tr>
<tr>
<td><strong>DescDetail</strong>a</td>
<td>The natural log of the number of characters in the project description.</td>
</tr>
<tr>
<td><strong>AcceptDonation</strong>b</td>
<td>Indicator variable that is 1 if the project has been set up to accept donations from users, 0 otherwise.</td>
</tr>
<tr>
<td><strong>CodeReleased</strong>c</td>
<td>Indicator variable that is 1 if the project has released code prior to the focal date, 0 otherwise.</td>
</tr>
<tr>
<td><strong>TeamSize</strong>c</td>
<td>The natural log of the number of developers (including project initiator) who are listed as project members prior to the focal date.</td>
</tr>
<tr>
<td><strong>DeveloperExpActiveT</strong></td>
<td>The number of months since the focal developer first became involved as a member of a project at SourceForge.net.</td>
</tr>
<tr>
<td><strong>DomainMatch</strong></td>
<td>Percentage of the developer’s prior OSS projects that have at least one domain in common with the focal project.</td>
</tr>
<tr>
<td><strong>PLMatch</strong></td>
<td>Percentage of the developer’s prior OSS projects that have at least one programming language in common with the focal project.</td>
</tr>
<tr>
<td><strong>AudienceMatch</strong></td>
<td>Percentage of the developer’s prior OSS projects that have at least one type of intended audience in common with the focal project.</td>
</tr>
<tr>
<td><strong>OSMatch</strong></td>
<td>Percentage of the developer’s prior OSS projects that have at least one type of operating system in common with the focal project.</td>
</tr>
<tr>
<td><strong>ProjectAge</strong>c</td>
<td>The natural log of the number of days since the project was registered at SourceForge.net until the focal date.</td>
</tr>
<tr>
<td><strong>PreexistingCode</strong>b</td>
<td>Indicator variable that takes 1 if the project’s earliest release date is prior to its registration at SourceForge.net, 0 otherwise.</td>
</tr>
</tbody>
</table>

**Notes:**
(a) All four tie variables are coded as 0 for developer-initiator or developer-developer pairs without prior ties.
(b) Time-invariant project attribute measures.
(c) Time-varying project attribute measures whose values depend on the date on which the focal developer joins the project, that is, even for developers joining the same project the values will depend on the actual join date. Daily data collection enabled this fine-grained measure operationalization.
Table 3. Domain Distribution of Sample Projects

<table>
<thead>
<tr>
<th>Category</th>
<th>Freq</th>
<th>Percentage</th>
<th>Category</th>
<th>Freq</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Software Development</td>
<td>276</td>
<td>20.32</td>
<td>Education</td>
<td>48</td>
<td>3.53</td>
</tr>
<tr>
<td>Internet</td>
<td>245</td>
<td>18.04</td>
<td>Security</td>
<td>42</td>
<td>3.09</td>
</tr>
<tr>
<td>System</td>
<td>206</td>
<td>15.17</td>
<td>Desktop Environment</td>
<td>29</td>
<td>2.14</td>
</tr>
<tr>
<td>Communications</td>
<td>166</td>
<td>12.22</td>
<td>Text Editors</td>
<td>25</td>
<td>1.84</td>
</tr>
<tr>
<td>Games/Entertainment</td>
<td>161</td>
<td>11.86</td>
<td>Other</td>
<td>17</td>
<td>1.25</td>
</tr>
<tr>
<td>Scientific/Engineering</td>
<td>154</td>
<td>11.34</td>
<td>Printing</td>
<td>8</td>
<td>0.59</td>
</tr>
<tr>
<td>Multimedia</td>
<td>136</td>
<td>10.10</td>
<td>Terminals</td>
<td>6</td>
<td>0.44</td>
</tr>
<tr>
<td>Office/Business</td>
<td>111</td>
<td>8.17</td>
<td>Religion and Philosophy</td>
<td>2</td>
<td>0.15</td>
</tr>
<tr>
<td>Database</td>
<td>66</td>
<td>4.86</td>
<td>Sociology</td>
<td>1</td>
<td>0.07</td>
</tr>
<tr>
<td>Formats and Protocols</td>
<td>62</td>
<td>4.57</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Notes: The total number of projects is 1358, which includes only those projects that have defined its domain during our data collection period. Also since projects can have multiple overlapping domains, the cumulative frequency is greater than 100%.

Table 4. Descriptive Statistics

<table>
<thead>
<tr>
<th>Variable</th>
<th>Mean</th>
<th>St. Dev</th>
<th>Min</th>
<th>Max</th>
</tr>
</thead>
<tbody>
<tr>
<td>Join</td>
<td>0.17</td>
<td>0.377</td>
<td>0.00</td>
<td>1.00</td>
</tr>
<tr>
<td>InitiatorTieOutcome</td>
<td>0.00</td>
<td>0.117</td>
<td>-2.53</td>
<td>2.27</td>
</tr>
<tr>
<td>InitiatorTieStrength</td>
<td>0.02</td>
<td>0.611</td>
<td>-0.50</td>
<td>42.90</td>
</tr>
<tr>
<td>OtherTieOutcome</td>
<td>0.00</td>
<td>0.120</td>
<td>-2.29</td>
<td>1.64</td>
</tr>
<tr>
<td>OtherTieStrength</td>
<td>0.02</td>
<td>0.349</td>
<td>-0.89</td>
<td>13.30</td>
</tr>
<tr>
<td>InitiatorTieAmount</td>
<td>0.48</td>
<td>1.058</td>
<td>0.00</td>
<td>5.84</td>
</tr>
<tr>
<td>OtherTieAmount</td>
<td>0.30</td>
<td>0.977</td>
<td>0.00</td>
<td>6.67</td>
</tr>
<tr>
<td>InitiatorExpActiveT</td>
<td>18.03</td>
<td>13.780</td>
<td>0.00</td>
<td>73.64</td>
</tr>
<tr>
<td>OtherExpActiveT</td>
<td>5.65</td>
<td>10.455</td>
<td>0.00</td>
<td>73.36</td>
</tr>
<tr>
<td>InitiatorExpP</td>
<td>0.40</td>
<td>0.596</td>
<td>0.00</td>
<td>3.06</td>
</tr>
<tr>
<td>OtherExpP</td>
<td>0.10</td>
<td>0.361</td>
<td>0.00</td>
<td>2.83</td>
</tr>
<tr>
<td>DomainPopularity</td>
<td>0.06</td>
<td>0.088</td>
<td>0.00</td>
<td>0.56</td>
</tr>
<tr>
<td>ActivityPercentile</td>
<td>2.19</td>
<td>0.489</td>
<td>1.00</td>
<td>3.00</td>
</tr>
<tr>
<td>DescDetail</td>
<td>4.97</td>
<td>0.633</td>
<td>1.73</td>
<td>5.62</td>
</tr>
<tr>
<td>AcceptDonation</td>
<td>0.08</td>
<td>0.270</td>
<td>0.00</td>
<td>1.00</td>
</tr>
<tr>
<td>CodeReleased</td>
<td>0.03</td>
<td>0.161</td>
<td>0.00</td>
<td>1.00</td>
</tr>
<tr>
<td>TeamSize</td>
<td>0.40</td>
<td>0.683</td>
<td>0.00</td>
<td>3.71</td>
</tr>
<tr>
<td>DeveloperExpActiveT</td>
<td>27.73</td>
<td>20.684</td>
<td>0.00</td>
<td>73.30</td>
</tr>
<tr>
<td>DomainMatch</td>
<td>0.02</td>
<td>0.131</td>
<td>0.00</td>
<td>1.00</td>
</tr>
<tr>
<td>PLMatch</td>
<td>0.06</td>
<td>0.236</td>
<td>0.00</td>
<td>1.00</td>
</tr>
<tr>
<td>AudienceMatch</td>
<td>0.13</td>
<td>0.322</td>
<td>0.00</td>
<td>1.00</td>
</tr>
<tr>
<td>OSMatch</td>
<td>0.08</td>
<td>0.267</td>
<td>0.00</td>
<td>1.00</td>
</tr>
<tr>
<td>ProjectAge</td>
<td>3.25</td>
<td>0.514</td>
<td>2.40</td>
<td>4.67</td>
</tr>
<tr>
<td>PreexistingCode</td>
<td>0.01</td>
<td>0.089</td>
<td>0.00</td>
<td>1.00</td>
</tr>
</tbody>
</table>
Alon, 2003) continuous and the other dichotomous, and the phi coefficient of correlation was used when both variables were dichotomous (Cohen, Cohen, West, and

Note: The Pearson correlation coefficient was used when both variables were continuous. The point-biserial correlation was used when one variable was

<table>
<thead>
<tr>
<th>(23)</th>
<th>(22)</th>
<th>(21)</th>
<th>(20)</th>
<th>(19)</th>
<th>(18)</th>
<th>(17)</th>
<th>(16)</th>
<th>(15)</th>
<th>(14)</th>
<th>(13)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.00</td>
<td>0.24</td>
<td>0.10</td>
<td>0.00</td>
<td>0.00</td>
<td>0.20</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.20</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

Table 2: Inter-Correlations

Significance keys: (4) $p < 0.01$, **$p < 0.05$, *$p < 0.1$
Table 6. Logistic Regression Results

<table>
<thead>
<tr>
<th>Variable</th>
<th>Parameter Estimate</th>
<th>95% Confidence Interval</th>
<th>Odds Ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Constant</strong></td>
<td>-8.119</td>
<td>(-10.442, -5.655)</td>
<td></td>
</tr>
<tr>
<td><strong>Impact of Cohesion Cues based on Direct Past Ties between Developer and Project Members (H1, H2)</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>InitiatorTieOutcome</td>
<td>0.27</td>
<td>(-2.289, 2.025)</td>
<td>0.973</td>
</tr>
<tr>
<td>InitiatorTieStrength</td>
<td>0.708</td>
<td>(0.164, 1.972)</td>
<td>2.030</td>
</tr>
<tr>
<td>OtherTieOutcome</td>
<td>0.985</td>
<td>(-2.585, 5.802)</td>
<td>2.678</td>
</tr>
<tr>
<td>OtherTieStrength</td>
<td>0.367</td>
<td>(-1.395, 3.289)</td>
<td>1.443</td>
</tr>
<tr>
<td><strong>Impact of Status Cues based on Project Members’ Prior Collaboration Experiences (H3)</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>InitiatorTieAmount</td>
<td>0.190</td>
<td>(-0.580, 0.206)</td>
<td>0.827</td>
</tr>
<tr>
<td>OtherTieAmount</td>
<td>0.384</td>
<td>(-0.742, -0.069)</td>
<td>0.684</td>
</tr>
<tr>
<td>InitiatorExpActiveT</td>
<td>0.014</td>
<td>(-0.042, 0.011)</td>
<td>0.986</td>
</tr>
<tr>
<td>OtherExpActiveT</td>
<td>0.027</td>
<td>(0.011, 0.043)</td>
<td>1.027</td>
</tr>
<tr>
<td>InitiatorExpP</td>
<td>0.009</td>
<td>(-0.504, 0.410)</td>
<td>0.991</td>
</tr>
<tr>
<td>OtherExpP</td>
<td>0.678</td>
<td>(0.036, 1.363)</td>
<td>1.970</td>
</tr>
<tr>
<td><strong>Additional Factors Affecting Developer Decisions to Join New Projects</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DomainPopularityₐ</td>
<td>0.569</td>
<td>(-1.565, 2.745)</td>
<td>1.766</td>
</tr>
<tr>
<td>ActivityPercentileLowₐ</td>
<td>-0.974</td>
<td>(-2.227, -0.176)</td>
<td>0.379</td>
</tr>
<tr>
<td>ActivityPercentileMidₐ</td>
<td>-0.099</td>
<td>(-0.462, 0.289)</td>
<td>0.906</td>
</tr>
<tr>
<td>DescDetail</td>
<td>0.091</td>
<td>(-0.250, 0.405)</td>
<td>1.095</td>
</tr>
<tr>
<td>AcceptDonation</td>
<td>0.655</td>
<td>(0.166, 1.145)</td>
<td>1.925</td>
</tr>
<tr>
<td>CodeReleased</td>
<td>1.280</td>
<td>(0.193, 2.077)</td>
<td>3.597</td>
</tr>
<tr>
<td>TeamSize</td>
<td>2.550</td>
<td>(2.167, 3.089)</td>
<td>12.807</td>
</tr>
<tr>
<td>DeveloperExpActiveT</td>
<td>-0.042</td>
<td>(-0.056, -0.031)</td>
<td>0.961</td>
</tr>
<tr>
<td>DomainMatch</td>
<td>-0.093</td>
<td>(-2.272, 1.441)</td>
<td>0.911</td>
</tr>
<tr>
<td>PLMatch</td>
<td>-0.142</td>
<td>(-1.263, 0.639)</td>
<td>0.868</td>
</tr>
<tr>
<td>AudienceMatch</td>
<td>-1.077</td>
<td>(-1.833, -0.510)</td>
<td>0.343</td>
</tr>
<tr>
<td>OSMatch</td>
<td>-0.774</td>
<td>(-1.690, -0.03)</td>
<td>0.463</td>
</tr>
<tr>
<td>ProjectAge</td>
<td>-1.782</td>
<td>(-2.316, -1.253)</td>
<td>0.169</td>
</tr>
<tr>
<td>PreexistingCode</td>
<td>0.093</td>
<td>(-1.330, 1.333)</td>
<td>1.097</td>
</tr>
</tbody>
</table>

Significance levels: * * * \( p < 0.01 \), * * \( p < 0.05 \), * \( p < 0.1 \)

Notes: (a) It is possible that DomainPopularity does not fully capture domain-specific effects. Therefore, we further checked the robustness of our findings by including dummy variables representing whether a project belongs to the top seven domains from Table 1. None of these variables has a significant effect on whether developers joined a project. Furthermore, we constructed 1000 bootstrap samples by sampling projects in the top five domains (\( N = 890 \)). Results are consistent with those shown.
(b) Since ActivityPercentile is a categorical variable (low, mid or high), we report the results using two dummy variables where “high” is the base case for comparison.