Leadership and learning in distributed teams: The case of Open Source Software Development

Abstract

Army training for soldiers and leaders must increasingly include preparation for operations in technology-mediated settings. We propose a sociological study in the context of distributed teams of software developers to identify theoretically-driven and empirically-grounded specifications of leadership and learning behaviors, capabilities and skills that critically influence distributed team performance. As well, we will experimentally validate antecedents of leadership and learning in these distributed teams, thus providing a basis for future training programs and other interventions to enhance distributed team performance. Our study addresses two general research questions: How does leadership emerge and function in distributed technology-mediated teams of software developers? and How do developers in these teams form shared mental models, informal norms and formal rules, and how do these structures guide their behaviours?

To answer these questions, we will conduct a longitudinal in-depth action research study describing and comparing the formation and evolution of distributed teams of Free/Libre Open Source (FLOSS) developers. FLOSS teams are extreme examples of bottom-up self-organization, being composed largely of volunteer members who interact via computer-mediated communications (CMC). We will study how these distributed groups develop shared mental models to guide members' behavior, roles to mediate access to resources and to provide leadership for the group, and norms and rules to shape action, as well as the dynamics by which independent, geographically-dispersed individuals are socialized into teams. As a basis for this study, we develop a structurational framework that integrates research on team behavior, organizational learning, communities of practice and shared mental models. We will utilize qualitative data analysis of team interactions, observation and interview data to investigate the team dynamics. We will also use social network analysis to study the change in communications and roles over time.

The study will have conceptual, empirical and practical contributions. Developing an integrated theoretical framework to understand the dynamics of learning and leadership in distributed teams is a contribution to the study of distributed teams. The project will contribute to advancing knowledge and understanding of FLOSS development and distributed work more generally by identifying how these teams evolve and how new members are socialized, thus filling a gap in the literature on self-organization and emergent leadership with an in-depth investigation based on a large pool of data. As well, we will use several different techniques to analyze the practices, providing different perspectives of analysis and a more reliable portrait of what happens in FLOSS teams.

Understanding these questions is of practical important because a network organization entails an increased use of distributed teams for a wide range of work. If successful, the project will shed light on learning and leadership for distributed work teams, which will be valuable for managers who intend to implement such an organizational form. The results of the study can serve as guidelines to behavior or as the basis for training (e.g., for team governance, task coordination, communication practices, mentoring) to improve distributed team organization and performance.

Table of Contents

Leadership and learning in distributed teams: The case of Open Source Software Development

Background

Army training for soldiers and leaders must increasingly include preparation for operations in technology-mediated settings. Implementation of network-centric warfare [2,3] demands that forces be able to organize from the bottom-up while connected only through technological links, which has significant implications for leadership and learning. Our research will identify theoretically-driven and empirically-grounded specifications of leadership and learning behaviors, capabilities and skills that critically influence distributed team performance. As well, we will experimentally validate antecedents of leadership and learning in distributed teams, thus providing a basis for future training programs and other interventions to enhance team performance, especially when leaders and members are separated by distance.

To address these issues, we propose a sociological study in the context of distributed teams of software developers to better understand the cognitive and social structures that underlie changes in individual and team behaviours in these teams. Software developers provide a useful setting for our research because the contributions of individual team members and their interactions are particularly visible. We will address two general research questions:

1. How does leadership emerge and function in distributed technology-mediated teams of Open Source Software developers?
2. How do developers in these teams form shared mental models, informal norms and formal rules, and how do these structures guide their behaviours?

To answer these questions, we propose a longitudinal in-depth action research study identifying and comparing the formation and evolution of distributed teams of software developers. We have chosen to situate our study in the real-world setting of Free/Libre Open Source Software (FLOSS) development. Revolutionary technologies and ideas have created a more closely linked world with almost instantaneous transmission of information to feed a global economy. A prominent example of this transformation is the emergence of FLOSS (e.g., Linux or Apache), software created by distributed dynamic teams of volunteers and professionals working in a loosely coupled fashion. FLOSS is a broad term used to embrace software developed and released under an "open source" license allowing inspection, modification and redistribution of the software's source without charge ("free as in beer"). Much (though not all) of this software is also "free software", meaning that derivative works must be made available under the same unrestrictive license terms ("free as in speech", thus "libre"). We have chosen to use the acronym FLOSS rather than the more common OSS to emphasize this dual meaning. There are thousands of FLOSS projects, spanning a wide range of applications. Due to their size, success and influence, the Linux operating system and the Apache Web Server (and related projects) are the most well known, but hundreds of others are in widespread use, including projects on Internet infrastructure (e.g., sendmail, bind), user applications (e.g., Mozilla, OpenOffice) and programming languages (e.g., Perl, Python, gcc). Many are popular (indeed, some dominate their market segment) and the code has been found to be generally of good quality [4].

Key to our interest is the fact that most FLOSS software is developed by distributed teams. Developers contribute from around the world, meet face-to-face infrequently if at all, and coordinate their activity primarily by means of computer-mediated communications (CMC) [5,6]. These teams depend on processes that span traditional boundaries of place and ownership. The research literature on software development and on distributed work emphasizes the difficulties of distributed software development, but the case of FLOSS development presents an intriguing counter-example. What is perhaps most surprising about the FLOSS process is that it appears to eschew traditional project coordination mechanisms such as formal planning, system-level design, schedules, and defined development processes [7,8]. As well, many (though by no means all) programmers contribute to projects as volunteers and without working for a common organization, making FLOSS teams extreme examples of bottom-up organizing. Studying such teams will expose work and coordination practices for distributed teams that require neither formal rank nor hierarchy.

As well, FLOSS development is an important phenomena deserving of study for itself. FLOSS is an increasingly important commercial phenomenon involving all kinds of software development firms, large, small and startup. Millions of users depend on systems such as Linux and the Internet (heavily dependent on FLOSS tools), but as Scacchi [9] notes, “little is known about how people in these communities coordinate software development across different settings, or about what software processes, work practices, and organizational contexts are necessary to their success”.

The remainder of this proposal is organized into four sections. In section 1, we present the research setting and discuss the challenges faced by FLOSS teams, which parallel those of other bottom-up organizations. In the technical approach, we first develop a conceptual framework for our study, drawing on theories of leadership and organizational learning [10,11], and using structuration theory [1] as an organizing framework. In the following section, we present an action research study design, along with details of the planned data collection and analysis plans. We conclude by sketching the expected contributions of our study.

The challenge of distributed FLOSS software development

The nascent research literature on FLOSS has addressed a variety of questions. First, researchers have examined the implications of FLOSS from economic and policy perspectives. For example, some authors have examined the implications of free software for commercial software companies or the implications of intellectual property laws for FLOSS [12-14]. Second, various explanations have been proposed for the decision by individuals to contribute to projects without pay [15-19]. These authors have mentioned factors such as personal interest, ideological commitment, development of skills [20] or enhancement of reputation [19]. Finally, a few authors have investigated the processes of FLOSS development [e.g., 5,21], which is the focus of this proposal.

Raymond’s [5] bazaar metaphor is perhaps the most well-known model of the FLOSS process. As with merchants in a bazaar, FLOSS developers are said to autonomously decide how and when to contribute to project development, that is, through bottom up self-synchronization. By contrast, traditional software development is likened to the building of a cathedral, progressing slowly under the control of a master architect (a top-down command and control model). While popular, the bazaar metaphor has been broadly criticized. According to its detractors, the bazaar metaphor disregards important aspects of the FLOSS process, such as the importance of project leader control, the existence of de-facto hierarchies and emergent leadership, the danger of information overload and burnout, and the possibility of conflicts that cause a loss of interest in a project or forking [22,23]. Clearly, a more theoretically grounded approach is needed to make progress in this area.

For the purposes of this study, we have chosen to analyze developers as comprising a work team. Much of the literature on FLOSS has conceptualized developers as forming communities, which is a useful perspective for understanding why developers choose to join or remain in a project. However, for the purpose of this study, we view the projects as entities that have a goal of developing a product, whose members are interdependent in terms of tasks and roles, and who have a user base to satisfy, in addition to having to attract and maintain members. These aspects of FLOSS projects suggest analyzing them as work teams. Guzzo and Dickson [24] defined a work team as "made up of individuals who see themselves and who are seen by others as a social entity, who are interdependent because of the tasks they perform as members of a group, who are embedded in one or more larger social system (e.g., community, or organization), and who perform tasks that affect others (such as customers or coworkers)".

More specifically, FLOSS projects are examples of distributed teams. Distributed teams are groups of geographically dispersed individuals working together over time towards a common goal. Though distributed work has a long history [e.g., 25], advances in information and communication technologies have been crucial enablers for recent developments in this organizational form [26]. Distributed teams seem particularly attractive for software development because the code can be shared via the systems used to support team interactions [27,28]. While distributed teams have many potential benefits, distributed workers face many real challenges. Watson-Manheim, Chudoba, & Crowston [29] argue that distributed work is characterized by numerous discontinuities: a lack of coherence in some aspects of the work setting (e.g., organizational membership, business function, task, language or culture) that hinders members in making sense of the task and of communications from others [30], or that produces unintended information filtering or misunderstandings [31]. These interpretative difficulties in turn make it hard for team members to develop shared mental models of the developing task [32,33].

While discontinuities are a pervasive problem for distributed teams, their presence seems likely to be particularly problematic for software developers [30]. Numerous studies of the social aspects of software development teams [30,34-37] conclude that large system development requires knowledge from many domains, which is thinly spread among different developers [34]. As a result, large projects require a high degree of knowledge integration and the coordinated efforts of multiple developers [38]. More effort is required for interaction when participants are distant and unfamiliar with each others work [39,40]. The additional effort required for distributed work often translates into delays in software release compared to traditional face-to-face teams [41,42]. The problems facing distributed software development teams are reflected in Conway's law, which states that the structure of a product mirrors the structure of the organization that creates it. Accordingly, splitting software development across a distributed team will make it hard to achieve an integrated product [7]. Such coordination problems have parallels in other applications of distributed cooperation.

Distributed FLOSS development teams also face problems in leadership. A lack of common knowledge about the status, authority and competencies of team participants can be an obstacle to the development of team norms [43] and conventions [44]. Leadership is one of the most studied topics in organizational and management research, and the study of technology-mediated collaboration in virtual teams is also widespread, but relatively little research has been conducted the nature of leadership in virtual, distributed teams [45,46]. One reason for this omission may be that our traditional notions of leadership do not seem easy to apply to distributed teams. Such teams are often composed of people of relatively equal status, or who are so disparate in background that formal organizational status seems irrelevant. Thus the usual leadership cues provided by organizational status and title are either absent or attenuated. Such groups often have no appointed leader, and their members may or may not have significant prior experience working with one another. These groups may formally appoint a single chairman or leader, but often this act is deferred or never actually occurs. In such cases a leader or leaders may emerge gradually, and such emergent leadership may be completely unrelated to organizational position or status. Effective emergent leaders may be those who are able to attend to both the relational (social) and task-related needs of the group, adapting to the situation and manifesting the requisite behaviors as required [47-51].

Virtual teams can either be self-organizing or formally organized by the organizations in which they are embedded. They can be hierarchical (with a single leader) or participative (with multiple leaders). This observation suggests that there may be several different forms of leadership appropriate to distributed teams. It is possible that leadership, like the teams themselves, can be both distributed and emergent. Thus the most effective types of leadership behavior in these new organizational forms might be very different than the behaviours appropriate to the centralized, hierarchical, single leader paradigm. Learner and Tirole [52] suggest important roles for leaders in FLOSS projects, including providing a vision for, coordinating and motivating contributors. However, Edwards [53] critiques their analysis, concluding that, “A study applying leadership theory in an open source context would indeed be revealing to the applicability of leadership in this special non-profit context.” Overall, research and practitioner communities know little about the processes of knowledge sharing, learning, socialization or leadership suitable for distributed teams [54,55].

In response to the problems created by discontinuities, studies of distributed teams stress the need for a significant amount of time spent learning how to communicate, interact, socialize and lead using computer-supported communications tools [56]. Research has shown the importance of formal and informal coordination mechanisms and information sharing [35] for a project’s performance and quality. Communication can help clarify potential uncertainties and ambiguities and socialize members with different cultures and approaches into a cohesive team [48,57-60]. Successful distributed teams share knowledge and information and create new practices to meet the task and social needs of the members [55]. However, the dynamics of knowledge sharing, socialization and leadership for distributed teams are still open topics for research [e.g., 54].

Technical Approach

Conceptual development

In this section we develop the conceptual framework for our study. We first briefly review relevant literature on leadership and on learning before presenting a structurational perspective of the dynamics of leadership and learning in FLOSS teams.

Leadership

There is an extensive literature on leadership, but much of does not comfortably apply to the context of distributed virtual teams. At the risk of oversimplifying, most leadership theories have tended to view the leader as a single, dominant individual (the "great man"), usually (though not always) occupying a formally defined leadership position in the social structure. Trait and style theories of leadership attempt to define the traits or behaviors that differentiate leaders from others [e.g., 61,62]. Situational and contingency theories of leadership attempt to identify the relationship between the individual leader's traits or behavioral style and the environmental circumstances in which the leader operates [63,64]. These approaches do not seem to provide much leverage for understanding leadership in bottom-up organizations.

We turn instead to literature adopting a functional approach to leadership, which focuses on the behaviors of a group rather than on a particular individual. In this approach some behaviors serve as leadership functions in that they help the group to achieve its goals and perform effectively. More than one individual may perform leadership behaviors, and different individuals may perform the same leadership behaviors at different times [65]. Thus a functional approach to leadership is better suited to the observation of emergent leadership behaviors in teams without *a priori* leadership status or assignments.

It is important for our purposes to also note that most functionalist theories make a broad distinction between *task* leadership behaviors and *group maintenance* leadership behaviors. The former are concerned with organizing, coordinating and performing the task(s) that constitute the group's primary work. The latter are concerned with maintaining group morale, motivation and communication. Bales [66] believed that the functions of task and maintenance behaviors are opposed, and that groups should strive to find a balance or equilibrium between them. The opposition between task and maintenance behaviors also suggested to Bales that it would be more likely that different people would emerge to perform task and maintenance roles [65]. In addition to the task and group maintenance functions which leadership must satisfy, Ancona and Caldwell [67] argued that there are also leadership functions involved with maintaining relations with individuals and groups outside the team.

Finally, in a distributed team where members make diverse knowledge contributions [68], it may be useful to distinguish between two types of task roles, *procedural* and *substantive*. *Procedural behaviors* are those involved in coordinating the group's work (scheduling, dividing labor, creating processes, etc.) while *substantive behaviors* are those that actually accomplish the group's work (idea generation, evaluation, integration, synthesis, etc.) Thus, leaders may exercise their influence by means of their substantive expertise as well as through their coordinating and directing activities.

In groups without leaders determined by formal appointment or hierarchical position, leadership is said to be *emergent*. Emergent leadership has been studied in FTF groups, and through this research we have learned about several aspects of leadership emergence that may prove relevant to virtual teams. As with other functionalist approaches, the emergent approach stands in contrast to “great man” theories of leadership. The emergent approach to leadership recognizes that it is through the interactions of the group that one or more individuals emerge to perform the leadership behaviors that the group requires. Researchers on emergent leadership have been interested in differences in the behaviors of emergent leaders and other group members, and in understanding the ways in which emergent leaders influence group actions.

A number of studies have explored the relationship between communication and leadership emergence. Many researchers have found that group members’ perceptions of leadership are very closely related to the quantity of communication initiated by each individual. Literature reviews [69,70] have reported that the amount of an individual's communication in a group correlates with group member judgments of leadership in the range of .5 to .7. It is interesting to note that while such high correlations have been found consistently with task leadership roles, the correlation between communication quantity and maintenance leadership roles is far lower (about .15). More specifically, several studies indicate that group members are more likely to be judged as leaders if their task leadership communications take the form of procedural leadership behavior as opposed to substantive leadership behavior [e.g., 71,72,73].

More recently, studies by Zigurs and Kozar [74] and Heckman et. al [75] found that technology in a virtual environment could assume some of the roles that leaders were previously expected to fill. Their results suggest that technology does not just assume task roles, but may also perform socio-emotional roles as well. By eliminating some roles entirely, and partially assuming or enriching others, technology can significantly realign the role structure, thus releasing group energy for other tasks and roles. For groups that are technologically skillful, this holds the promise that the functions of leadership may be accomplished through behaviors that are distributed and emergent in ways that are unlikely or impossible in co-located teams. This notion is consistent with the behavioral complexity theory of Kayworth and Leidner [49], who argue that effective virtual teams demand a higher level of leadership behavioral complexity than do traditional teams.

Learning

The second conceptual foundation of our study is organizational learning. Scholars and practitioners alike have recognized that an organization’s capacity to learn is a core competency necessary for survival and competition in the complex distributed knowledge-based economy [10,76]. An organization “learns” by integrating the knowledge of members into its structure (products, rules, procedures, shared mental models, norms). The more adept an organization is at learning, the better it can be at adapting to the environment, correcting for error, and innovating [77]. As the dependencies between team members increase and the team is responsible for a collective outcome, this learning has to occur at the team as well as at the individual level [10,76,78,79]. Fundamental to facilitating team learning is a strong conceptualization of how teams learn. We use Huber’s [11] definition, which states that “An entity learns if… the range of its *potential behaviors is changed*.”

To conceptualize a team’s behavioral potential, we draw on Grant’s [68] knowledge-based view of the firm. Similar to Swieringa and Wierdsma’s [80] conceptualization of organizations, we conceptualize teams as social entities that hold implicit and explicit rules that guide individual members’ interpretation, contribution and behavior. Implicit rules are team norms and common understanding (or shared mental models). Explicit rules are team rules, policies, procedures, and requirements. Both implicit and explicit rules are mechanisms that exist on the team level, extraneous to the individual member level. The knowledge-based view of the firm suggests that a team creates such implicit and explicit rules by integrating the knowledge of its members [68]. A team creates coordination mechanisms, in the form of procedures and norms, to economize on communication, knowledge transfer and learning, thus reserving team decision making and problem solving for complex and unusual tasks [68].

Based on the two complementary views above, we argue that changes in the behavioral potential (cognitive and social structures) of a team will be observable through changes in explicit and implicit rules [81]. These changes in explicit and implicit rules are the result of integrating the knowledge of members into the team's structure reflecting potential behavioral changes within a team over time, what March et al. [82] and Hayes and Allison [81] refer to as learning on the group level.

In summary, similar to Urch Druskat and Kayes [83], we define team learning as the process by which team members share their knowledge and information and integrate it into the team's implicit and explicit rules leading to changes in the behavioral potential of the team. We operationalize team learning as the change in explicit (formal rules, procedures, structures and roles) and implicit rules (informal structure, norms), and shared mental models. For the purpose of this study we use a structurational perspective to investigate the processes that change those cognitive and social structures of teams in distributed environments.

A structurational perspective on team dynamics

To conceptualize the dynamics of FLOSS teams and the process of changes within them, we adopt a structurational perspective. Numerous authors have used a structurational perspective to support empirical analyses of group changes [84-88]. A discussion of the merits of each use is beyond the scope of this application. Here, we build on the view of structuration presented by Orlikowski [85] and Barley and Tolbert [1].

Structuration theory [89] is a broad sociological theory that seeks to unite action and structure and to explain the dynamic of their evolution. We chose this framework because it provides a dynamic view of the relations between team and organizational structures and the actions of those that live within, and help to create and sustain, these structures. The theory is premised on the duality of structures, that is, systems of signification, domination and legitimation that influence individual action. In this view, structure is recursive: the structural properties of a social system are both the means and the ends of the practices that constitute the social system. As Sarason [90] explains, in structuration theory:

> "The central idea is that human actors or agents are both enabled and constrained by structures, yet these structures are the result of previous actions by agents. Structural properties of a social system consist of the rules and resources that human agents use in their everyday interaction. These rules and resources mediate human action, while at the same time they are reaffirmed through being used by human actors or agents." (p. 48).

More simply put, by doing things, we create the way to do things.

By relating structure and function across time, structuration theory provides a framework for understanding the evolution of a team [91]. Barley and Tolbert [1] note that structuration is "a continuous process whose operations can be observed only through time" (p. 100). Figure 1, adapted from [1] shows the relation between institution (which the authors use synonymously with structure) and action, and how both evolve over time. In this figure, the two bold horizontal lines represent "the temporal extensions of Giddens' two realms of social structure: institutions and action," while the "vertical arrows represent institutional constraints on action" and the diagonal arrows, "maintenance or modification of the institution through action" (p.100). As Cassell [92] says, “to study the structuration of a social system is to study the ways in which that system, via the application of generative rules and resources, in the context of unintended outcomes, is produced and reproduced through interaction” (p. 119). Thus, our analysis will describe current team practices (the lower arrow) and current team structures (the upper arrow) and how these interact (the vertical and diagonal arrows) and change over time. In order to explain how the teams are evolving, we present the changes as states or stages (e.g., T1, T2 and T3 in the figure) and highlight the “dislocation of routines” and other temporal disruptions that lead to these different states [91].

Existing structures and the process of structuration

The structuration perspective also makes clear the importance of any initial structures that individual team members bring from prior experiences or from an external context (i.e., from an unseen T0 in the figure). Barley and Tolbert [1] note that “actors are more likely to replicate scripted behaviours” than to develop new ones. Orlikowski and Yates [93] argue similarly, suggesting that in a new situation individuals will typically draw on their existing repertoires of actions, reproducing those they have experienced as members of other communities. These prior experiences will provide an initial set of structures that guide behaviours, which will be particular important during the formative stages of the team.

The importance of prior structures is reinforced by other research. For example, Hackman’s [94] model of group performance suggests organizational context as an important factor affecting team processes. Finholt and Sproull [95] found that teams who do not work within a specific organizational context have a greater need for team learning. These results have been also been supported by our initial interviews with FLOSS developers, who see corporate participation having an effect on team processes and activities.

Conceptualizing structuration in FLOSS teams

To apply structuration as a perspective to conceptualize the dynamics of distributed FLOSS teams, we first must clarify the types of rules and resources that comprise the structure. For this work, we specifically consider three kinds of rules and resources that are “encoded in actors’ stocks of practical knowledge” [1] in the form of interpretive schemes, resources, and norms [1,96]. In the remainder of this section, we elaborate each of these three aspects of structure as they apply to FLOSS development in particular.

Interpretive schemes and structures of signification. Individual actors’ interpretive schemes create structures of *signification* and thus influence (and are created by) individual actions. To

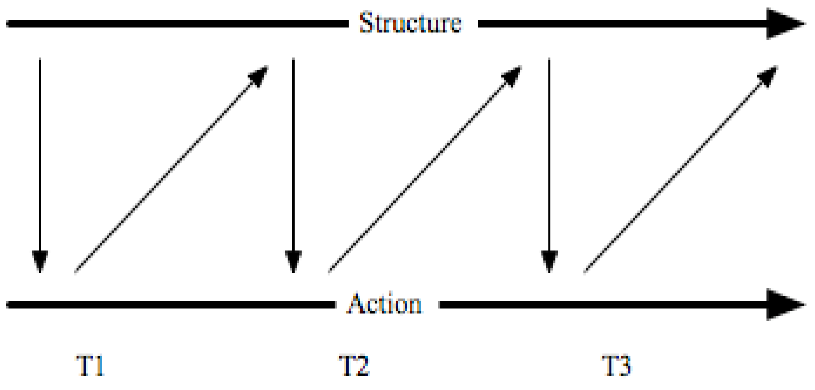


Figure 1. A sequential model of the relation between structure and action [from 1].

describe how these schemes influence action and vice versa, we draw on the literature on the role of shared mental models in team action. Shared mental models, as defined by Cannon-Bowers et al. [97], "are knowledge structures held by members of a team that enable them to form accurate explanations and expectations for the task, and in turn, to coordinate their actions and adapt their behavior to demands of the task and other team members" (p. 228). Without shared mental models, individuals from different teams or backgrounds may interpret tasks differently based on their backgrounds, making collaboration and communication difficult [98]. The tendency for individuals to interpret tasks according to their own perspectives and predefined routines is exacerbated when working in a distributed environment, with its more varied individual settings.

Research on software development in particular has identified the importance of shared understanding in the area of distributed software development, as in the case of FLOSS teams [99]. Curtis et al. [32], note that, "a fundamental problem in building large systems is the development of a common understanding of the requirements and design across the project team." They go on to say that, "the transcripts of team meetings reveal the large amounts of time designers spend trying to develop a shared model of the design". The problem of developing shared mental models is likely to be particularly affect FLOSS development, since FLOSS team members are distributed, have diverse backgrounds, and join in different phases of the software development process. In short, shared mental models are important as guides to effective individual contributions to, and coordination of the software development process.

In emphasizing the duality of structure, the structurational perspective draws our attention to how shared mental models are products of, as well as guides to, action. Walton and Hackman [100] identify an interpretive function of teams, which is to help members create a consistent social reality by developing shared mental models. To identify specific actions that can help to build shared mental models, we turn to Brown and Duguid [101], who identify the importance of socialization, conversation and recapitulation. First, new members joining a team need to be socialized into the team to understand how they fit into the process being performed. They need to be encouraged and educated to interact with one another to develop a strong sense of "how we do things around here" (e.g., norms). Barley and Tolbert [1] similarly note that socialization frequently "involves an individual internalizing rules and interpretations of behavior appropriate for particular settings" (p. 100). Second, conversation is critical in developing shared mental models. It is difficult to build shared mental models if people do not talk to one another and use common language. Meetings, social events, hallway conversations and electronic mail or conferencing are all ways in which team members can get in touch with what others are doing and thinking. Finally, Brown and Duguid [101] stress the importance of recapitulation. To keep shared mental models strong and viable, important events must be "replayed", reanalyzed, and shared with newcomers. The history that defines who we are and how we do things around here must be continually reinforced, reinterpreted, and updated.

While it might first appear that a consideration of leadership is more relevant to an understanding of the structures of domination than of signification, we expect it to play an important role in all three systems of structuration: signification, domination and legitimation. Our functional conceptualization of leadership as a set of distributed and emergent behaviors helps us to better understand the process of socialization through which shared mental models are developed. Social, or group maintenance leadership behaviors may be performed by different team members at different times, and can be expected to be distinguishable from, yet complement task leadership [46].

Most studies on shared mental models remain conceptual [102]. The few empirical studies [e.g., 99,103] investigated the relationship between team or organizational factors and the presence of shared mental models. This study will investigate the process through which distributed teams develop shared mental models. This will be accomplished through the analysis of interaction data for evidence of conversations, recapitulation of implicit and explicit rules and ideas about task, team members, attitudes, and beliefs.

Resources and structures of domination. Structures of *domination* emerged to ensure coordination and cooperation in groups to achieve objectives [68]. Coordination and cooperation depend on the exchange of the specialized knowledge between the various levels and kinds of expertise in an organization. The control of resources is the basis for power and thus for structures of *domination*. Resources include both allocative resources (control over things) and authoritative resources (control over people). For software development, material resources would seem to be less relevant, since the work is intellectual rather than physical and development tools are readily available, thanks to openly available FLOSS development systems such as SourceForge (http://sourceforge.net/) and Savannah (http://savannah.gnu.org/). Furthermore, most FLOSS teams have a stated ethos of open contribution. However, team members face important differences in access to expertise and control over system source code in particular. To understand the role of these sorts of resources, we plan to examine different roles in the software development process and how they affect individual contributions, and how these roles are established and maintained.

Several authors have described FLOSS teams as having a hierarchical or onion-like structure [104-106], as shown in Figure 2. At the centre are the core developers, who contribute most of the code and oversee the design and evolution of the project. Core developers are distinguished by having write privileges on the source code. The core is usually small and exhibits a high level of interaction, which would be difficult to maintain if the core group were large. Surrounding the core are co-developers. These individuals contribute sporadically by reviewing or modifying code or by contributing bug fixes. The co-developer group can be much larger than the core, because the required level of interaction is much lower. Surrounding the developers are the active users: a subset of users who use the latest releases and contribute bug reports or feature requests (but not code). Still further from the core are the passive users. The border of the outer circle is indistinct because the nature and variety of FLOSS distribution channels makes it difficult or impossible to know the exact size of the user population.

As their involvement with a project changes, individuals may move from role to role. For example, a common pattern is for active users to be invited to join the core development team in recognition of their contributions and ability. In some teams, this selection is an informal process managed by the project initiator, whiles others have formal voting processes for vetting new members. However, core developers must have a deep understanding of the software and the development processes, which poses a significant barrier to entry [107,108]. This barrier is particularly troubling because of the reliance of FLOSS projects on volunteer submission and "fresh blood" [109]. These characteristics again emphasize the importance of socialization and movement of individuals through roles in the projects. They also illustrate the way in which behaviors demonstrating substantive expertise constitute a form of leadership.

Rules and norms and structures of legitimation. Rules and norms are instruments of coordination and control that "routinize organizational activities and define authority relations, connections among subunits, and decision-making structures" [82]. Actors' social norms and team rules embody structures of *legitimation*. The regulative function of teams, as presented by Walton and Hackman [100], describes one aspect of team functions as the creation of rules, implicit and explicit. To conceptualize this aspect of teams, we also draw on Swieringa and Wierdsma's [80] description of organizations as collections of implicit and explicit rules that guide member behaviours. Implicit rules are team norms, shared amongst members of the team. Explicit rules are the stated rules, policies, procedures and team requirements defined for the team. We are particularly interested in the way these rules guide individual contributions to the team's goals.

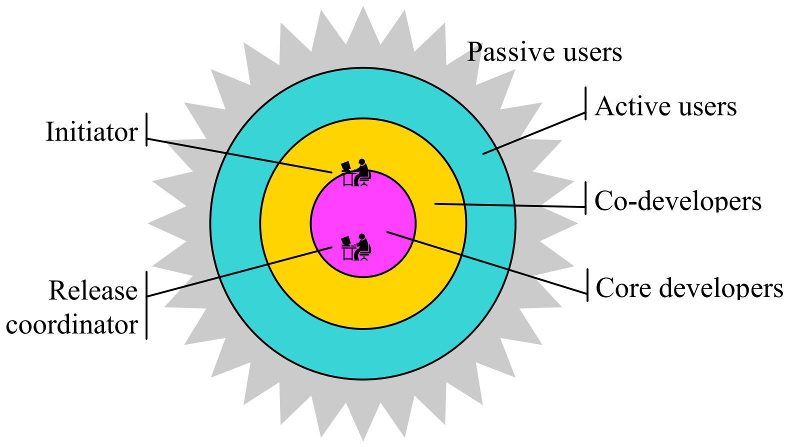


Figure 2. Hypothesized FLOSS development team structure.

As the team attempts to achieve its task, team interactions lead to the development of implicit and explicit rules for social or interpersonal interaction to guide team member behavior in achieving its goals and functions. These changes are the result of integrating the knowledge of experts (through problem solving, political negotiation, and experiential learning [82] into the team's structure reflecting changes to potential behavioral within a team over time. The creation and implementations of rules is a key competency for any group or organization [82]. A group or organization's ability to creatively create rules that are consistent with members' actions and represent organizational mission, values and process is critical to its effectiveness [82,110]. They also reflect what we have labeled procedural task leadership at work. The member or members of a team who initiate the coordinating processes that result in the development of implicit or explicit rules are those most likely to be perceived as leaders by other members [72,73].

Summary

Combining the discussion of the three aspects of structure described above results in the conceptual framework shown in Table 1. For each of the three aspects of structure, the table describes the embodiment of the structure as we have conceptualized it for FLOSS teams, and the actions that are guided by the structures and that reinforce or modify the structures. The resulting model is largely consistent with Grant's knowledge-based view of the firm [68], which analyzes

Table 1. Constructs for study: Embodiments of structures and actions that reinforce or modify structures.

Structure	**Structural embodiment**	**Actions that create/ reinforce/modify structure**
Signification	Shared mental models	Socialization Conversation Recapitulation
Domination	Roles with differential access to resources Leadership	Role definition Role assignment
Legitimation	Norms Formal rules and procedures	Rule creation and change

a firm as a structure for integrating specialist knowledge into the firm's activities and products [68]. Though this theory was originally stated in terms of firms, it is easily applicable to FLOSS development and other distributed teams. The knowledge-based view presents coordination, shared mental models, communication and decision-making and learning as interdependent issues affecting the effectiveness of distributed teams. Grant suggests that to integrate knowledge, firms need coordination mechanisms including rules, sequencing and routines that economize on communication, knowledge transfer and learning, and team decision making and problem solving for the most complex and unusual tasks. Finally, although there is differentiation between experts in what they know, Grant identifies shared mental models as an important prerequisite for knowledge integration.

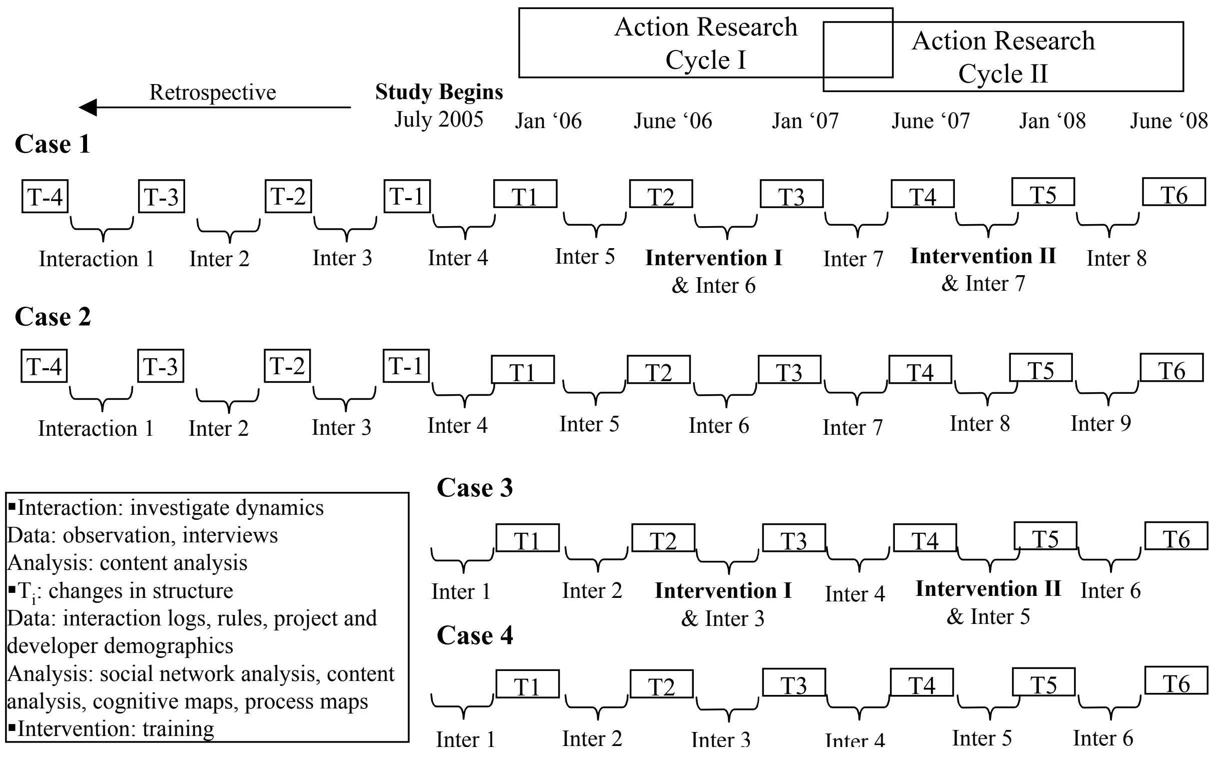
Research Design

In this section, we will discuss the design of the proposed study, addressing the basic research strategy (action research and longitudinal case studies), concepts to be examined, sample populations and proposed data collection and analysis techniques. We first discuss the goals and general design of the study. We then present the details of how data will be elicited and analyzed.

Longitudinal action-research study of four FLOSS teams

To study the dynamics of the leadership and learning of distributed teams of FLOSS developers, we will carry out a longitudinal action research study [111,112]. Action research is an interventionist approach to the development of scientific knowledge (in contrast to more typical observational or survey approaches). To understand the process of leadership and learning within each project and to create useful knowledge about how to improve the groups, researchers will act both as observers of the processes and as change agents in the projects, "alternating the

Figure 3. Research design.



▪Interaction: investigate dynamics
Data: observation, interviews
Analysis: content analysis
▪T_i: changes in structure
Data: interaction logs, rules, project and developer demographics
Analysis: social network analysis, content analysis, cognitive maps, process maps
▪Intervention: training

change agent and researcher roles" [113]. To implement the observer role, we will perform in-depth case longitudinal case studies of four FLOSS projects, following a four-step process suggested by Barley and Tolbert [1]. To implement and learn from the change agent role, we will follow a five-stage process suggested by Susman and Evered [111]. These stages and steps are interwoven, resulting in the overall research design shown in Figure 3.

<u>Longitudinal case studies.</u>

To develop an understanding of the processes of leadership and learning, we will build exploratory case studies of FLOSS groups [114]. As Yin defines it, a case study is "an empirical inquiry that investigates a contemporary phenomenon within its real-life context; when the boundaries between phenomenon and context are not clearly evident; and in which multiple sources of evidence are used" [115]. For some of the cases (cases 1 and 2 in Figure 3), we will combine the longitudinal study with retrospective data analysis. In this phase, we seek to build a theory of a complex organizational process comprising numerous small decisions and actions undertaken over years. To build this theory, we required detailed information about individuals' actions and attitudes. Therefore, qualitative research is more suited to explore the nature of the process of learning and leadership. We will examine the development of leadership and the learning process of FLOSS teams by investigation shared mental models, roles, rules and norm. Each case study will draw on multiple sources of data, including observation and participant observation, project and developer demographics, project plans and procedures, and interviews. The data will be analyzed using content analytic techniques, cognitive maps, process maps and social network analysis.

Barley and Tolbert [1] suggest the following four steps to investigate the dynamics of social and cognitive structures:

> "(1) defining an institution (structure) at risk of change over the term of the study and selecting sites in light of this definition; (2) charting flows of action at the sites and extracting scripts characteristic of particular periods of time; (3) examining scripts for evidence of change in behavioral and interaction patterns; and (4) linking findings from observational data to other sources of data on changes in the institution of interest" (pg. 103).

In the remainder of this section, we will discuss how we implement each of these steps, while deferring discussion of the details of data collection and analysis to subsequent sections.

Step one: Selecting sites. We will start by identifying promising projects for investigating the dynamics of structure and action. We plan to study four FLOSS project teams in depth to allow. In selecting teams to study, we will consider theoretical and pragmatic aspects.

- First, we will compare two newly-formed and two well-established project teams. We will study the development of the teams longitudinally and the two established teams retrospectively as well. Picking newly-formed teams will allow us to study the initial stages of team formation and in particular the negotiation among previously experienced structures brought in by team members. However, relying entirely on new teams seems risky. First, Barley and Tolbert [1] note the difficulties of identifying settings that are likely to experience interesting changes. Second, we want to ensure that we study some teams that have developed effective work practices. Studying some established teams allows us to choose some projects that are known to be effective. Studying established projects also permits study of the processes of socialization of new members into an ongoing project.

- Second, in order to ensure that we are studying genuine teams (as opposed to single person development efforts [116]), we will choose only projects with more than seven core developers, a lower limit for team size suggested by Hare [117].
- Finally, in selecting projects, we will also have to take into consideration some pragmatic considerations. We will select FLOSS teams where we have access to the data we need (e.g., message logs) and where we can obtain the participation of developers for interviews.

Step two: Charting flows of actions. In this step, we extract the interactions of team members within a particular time period to investigate the dynamics by which the teams develop rules, roles, shared mental models and norms and leadership emerges over time. We plan to interview developers for each case at least every six months (see Figure 3). Six months was chosen since it provides a small enough gap to be able to trace the process of change relying on developers' memories of events, while still being feasible for data collection and not too onerous for participants. We will also extract team interactions from email logs, ethnographic field notes, and observations of developer activities between the six month measurement points to analyze the dynamics that lead to the observed changes. For two of the cases, we will carry out a similar analysis on retrospective data (potentially over the entire recorded history of the project). The details of data elicitation and analysis are discussed in the following sections.

Step three: Identifying patterns of changes. Once we extract the segments of interactions discussed in step two, we will analyze the interaction to uncover the dynamics by which the teams develop their cognitive and social structures. More specifically we look to uncover the patterns of behavior through which members change shared mental models, roles, norms and rules and leadership emerges. We investigate the dynamics by which teams develop shared mental models by studying how members contribute to and coordinate tasks paying special attention to evidence of recapitulation, socialization, conversation. We study how role are assigned and evolve over time by studying member contribution and looking for evidence of role definition and role changes. Lastly, we study the dynamics by which rules and norms evolve by also looking for task contribution and coordination, paying special attention to evidence of rules creation and modification through problem solving, political negotiation, and experiential learning [82].

Step four: Linking changes in structures to other changes. In Step 4, Barley and Tolbert [1] suggest linking changes in the structures to other changes of interest in the sites being studied. Since the primary focus of our study is the dynamics of the teams, this step will not be the major focus of our efforts. Nevertheless, we will triangulate evidence gathered from multiples sources of evidence about the teams. For example, comparisons across the teams will provide evidence to help us understand the role of corporate participation in the teams. As well, we will use an action research approach to explore and validate the antecedents of change.

Action research

The research approach described above will provide reliable knowledge about the nature of the processes of leadership and learning. However, to develop an understanding of the antecedent of these processes requires that the researchers take a more active role. The situated nature of the phenomena of interest, leadership and learning, means that laboratory or field experiments are unlikely to provide sufficient external validity. Instead, we adopt an action research approach, in which researchers actively partner with participants in real world settings to explore and test theoretically motivated interventions.

Susman and Evered [111] describe a five-phase cyclic process for action research, consisting of 1) diagnosing, 2) action planning, 3) action taking, 4) evaluating and 5) specifying learning. Diagnosing includes identification of the primary problems that underlie the project's desire to change and leads to the development of working hypotheses about the state of the project. In this phase, we will develop rich descriptions of the dynamics and processes of the FLOSS projects, as described above.

In the next phase, action planning, researchers and practitioners collaborate in determining activities to address the problems identified. This planning is based on the theories and models brought to bear by the researchers as well as the knowledge of the practitioners. In other words, the research may be both theory-driven and theory-building. We anticipate interventions related to the antecedents of leadership and learning as discussed in the conceptual development section.

In the action-taking phase, the planned changes are implemented. For example, the research team may provide feedback to members of the FLOSS teams or even training on specific aspects of leadership behavior. We plan to carry out these development activities in two of the four projects, retaining the other two as controls. Being part of the change process enables the researchers to be participant-observers in the processes being studied. After the actions are taken, researchers and practitioners collaborate in evaluating the outcomes. This evaluation includes determining whether the actions had the theoretically expected effects and if they were effective in relieving the problems, a form of theory testing.

In the final phase, learnings from the actions and results are formally specified. This phase distinguishes action research as a research method rather than simply a type of change effort. Baskerville and Wood-Harper [118] suggest three audiences for these learnings. Firstly, the participant organizations can be restructured to reflect the new knowledge gained in the interaction. Secondly, where the change was unsuccessful or only partly successful, the learnings may lead to a new round of diagnosis and action planning. Finally, the test or building of the theoretical framework in practice contributes to the development of scientific knowledge.

Data collection

To explore the concepts identified in the conceptual development section of this proposal (Table 1), we will collect a wide range of data: project demographics, developer demographics, interaction logs, project plans and procedures, developer interviews, and project observation. In the remainder of this section, we will briefly review each source. Table 2 shows the mapping from each construct to data source.

Developer demographics. We will collect basic descriptive data about developers, such as area of expertise, formal role, years with the project, other projects the developer participates in etc. Often these data are self-reported by the developers on project pages; in other cases, they can be elicited from the developers during interviews. We will track changes in the formal rules and formal roles of members using this source.

Project plans and procedures. Many projects have stated release plans and proposed changes. Such data are often available on the project's documentation web page or in a "status" file used to keep track of the agenda and working plans [109]. For example, Scacchi [9] examined requirements documentation for FLOSS projects. We will also examine any explicitly stated norms, procedures or rules for taking part in a project, such as the process to submit and handle bugs, patches or feature request. Such procedures are often reported on the project's web page (e.g., http://dev.apache.org/guidelines.html). We will track changes in the various versions of any specific set of rules and procedures, roles and documented norms.

Interaction logs. The most voluminous source of data will be collected from archives of CMC tools used to support the team's interactions for FLOSS development work [42,119]. These data are useful because they are unobtrusive measures of the team's behaviours [120]. Mailing list archives will be examined, as email is a primary tool used to support team communication, learning and socialization. Such archives contain a huge amount of information: e.g., the Linux kernel list receives 5-7000 messages per month, the Apache httpd list receives an average of 40 messages a day. From mailing lists, we will extract the date, sender and any individual recipient' names, the sender of the original message, in the case of a response, and text of each message. We will examine features request archives and logs from other interaction tools, such as chat sessions. While in most cases these archives are public, we plan to consult with the Syracuse University Human Subjects Institutional Review Board to determine what kind of consent should be sought before proceeding with analysis. Mailing list archives is the main source of interaction data that illuminates the 'scripts' for the analysis of how these teams develop roles, rules, norms, and shared mental models and how leadership emerges [1]. Observation data from email logs can potentially provide a rich description of the behaviors (patterns of interaction) of FLOSS teams. This rich description leads to a better understanding of the processes of FLOSS development.

Observation. We have found from our initial pilot study (described below under Results from Prior Funding) that developers interact extensively at conferences. Indeed, Nardi and Whittaker [121] note the importance of face-to-face interactions for sustaining social relations in distributed teams. The FreeBSD developer Poul-Henning Kamp has also stated that phone calls can be occasionally used to solve complex problems [122]. These interactions are a small fraction of the to-

Table 2. Constructs, sources of data, and analysis.

Structure	**Constructs**	**Data sources (see section 3)**
Signification	Shared mental models	Content analysis of interactions, interviews and observation
	Task coordination and contribution	Process mapping, social network analysis
	Socialization Conversation Recapitulation	Content analysis of interactions, interviews and observation
Domination	Roles with differential access to resources	Process mapping, social network analysis Content analysis of interactions, interviews and observation
	Task coordination and contribution	(See above)
	Emergent leadership	Developer interviews and surveys
	Role definition Role changes	Process mapping, social network analysis
Legitimation	Norms Formal rules and procedures	Content analysis of interactions, interviews and observation Project plans and procedures
	Task coordination and contribution	(See above)
	Rule creation and change	Content analysis of interactions, interviews and observation

tal, but they may still be crucial to understanding the team's practices. We plan to use attendance at developer conferences as an opportunity to observe and document the role of face-to-face interaction for FLOSS teams.

Participant observation. We plan to carry out a virtual ethnographic study of developer socialization and interaction relying on participant observation of the teams. One student involved with the project has already virtually joined several development teams (with the permission of the project leaders and the knowledge of other members) and is currently participating in their normal activities while observing and recording these activities (following a protocol approved by the Syracuse University Human Subjects Review Board). In this way, we will study and learn first hand the socialization and coordination practices of these teams. We will track these teams through the various stages of development status, from planning through production/stable stage, observing how new members join the teams and how they contribute to the team output.

Developer interviews. While the data sources listed above will provide an extensive pool of data, they are mostly indirect. Interviews are important to get rich, first-hand data about developers' perceptions and interpretations. We plan to conduct interviews with key informants in the selected projects. Interviews will be conducted in part by e-mail, but we also plan to attend one or two FLOSS conferences each year (e.g., the *O'Reilly Open Source Convention* or *ApacheCon*) to interview FLOSS developers face-to-face. The first round of interviews will be scheduled after the initial data analysis to ensure that we have a sufficient understanding of the process to be able to pose intelligent questions, and on a recurring basis to provide insight into the dynamics of the team, as discussed above. We will explore the developer's initial experiences of participation in FLOSS, the social structure and norms of the team, processes of knowledge exchange and socialization (especially the role of observation or lurking, which leaves no traces in the interaction logs), and knowledge of other members' participation [123,124]. As well, interviews will be used to verify that the archives of interaction data give a fair and reasonably complete record of day-to-day interactions.

Developer survey. Surveys will be used to validate findings across the population of developers in each project. For ease of administration, the questionnaire will be administered via the Web. Fortunately, all members of the target population can be assumed to have Internet access and to be comfortable with the use of Web, so this choice of administration should not create any sampling biases.

Data analysis

While voluminous, the data described above are mostly at a low level of abstraction. The collected data will be analyzed using a variety of techniques in order to raise the level of conceptualization to fit the theoretical perspectives described in Section 2 and to address our research questions. Table 3 shows the mapping from data sources to data analysis techniques.

Content analysis. The project will rely heavily on content analysis of the text in interaction archives and interviews to develop insights on the extent and development of shared mental models and socialization (e.g., the way projects are created, introduction of new members, members leaving and community building). Data will be analyzed following the process suggested by Miles and Huberman [125], iterating between data collection, data reduction (coding), data display, and drawing and verifying conclusions. The researchers will develop an initial content analytic framework to discover the patterns of the various variables present in the data. The initial (deductive) framework will be based on indicators from content analytic frameworks previously used to investigate shared mental models [e.g., 126]. In addition we will incorporate

Table 3. Data sources and planned analysis approaches.

Data source	**Analysis approach**
Developer demographics	Statistical
Developer interaction logs	Social network analysis
	Content analysis, process mapping
Project plans and procedures	Content analysis
Developer interviews	Content analysis, process mapping, cognitive mapping
Observation of developer interactions	Content analysis, process mapping, cognitive mapping
Participant observation	Content analysis, process mapping, cognitive mapping
Developer survey	Statistical

coding schemes from our work investigating social, cognitive and structuring processes of virtual teams in the context of Asynchronous Learning Networks [127]. We will start the data analysis using those initial content analytic scheme and modify the scheme as new categories and indicators emerge in the data [125]. Further categories will be added and other data will be collected as preliminary findings in the analysis suggest. We will use the thematic unit of analysis while conducting the content analysis to capture the various elements of the variables under investigation as appropriate. To increase the validity and reliability of the coding scheme we will conduct intercoder reliability tests and modify the content analytic scheme until we reach an 85% agreement level [128].

Social network analysis (SNA). SNA will be used to analyze patterns of interactions (e.g., who responds to whose email) in order to reveal the structure of the social network of projects. Madey, Freeh & Tynan [129] applied this technique to connections between projects, but not within projects. We are particularly interested in using social network information to identify various structural roles in the team and how individuals fill these roles over time. This analysis of structural roles should provide a useful counterpoint to descriptions of formal roles. As well this analysis will track the socialization of members into the core of the team, and the development and changes in leadership over time. We will assess an individual's centrality and the project's hierarchy, which seems to mediate the effect of role and status on individual performance within virtual teams [26], the way contributions are distributed among developers and the roles assumed by core developers. The results of such analyses will support us in the identification of the social relations patterns and the way such patterns develop and affect team learning and socialization.

Process maps. The open source software development *processes* will be mapped based on an inductive coding of the steps involved. For example, to map the bug fixing process, we will examine how various bugs were fixed as recorded in the bug logs, email messages and the code. Van de Ven and Poole [130] describe in detail the methods they used to develop and test a process theory of how innovations develop over time. Yamauchi et al. [131] coded messages to understand the development processes of two FLOSS projects. Process traces can be clustered using optimal matching procedures [132] to develop clusters of processes. These process descriptions can be enriched with descriptions of the process from developers' reports of critical incidents and of the process in general [133].

In our analyses, we will identify which individuals perform which activities to identify different process *roles*, thus providing a counterpoint to the SNA roles described above. We will also identify the coordination modes and task assignment practices involved in software maintenance (i.e., the number of features request assigned, types of requests, number and types of spontaneous contributions), the adoption of other formal coordination modes (from the analysis of the written policies regarding contributions to projects), as well as the degree of interdependency among the tasks (based on an analysis of communication patterns among different roles and different contributors). Another question we intend to answer is the extent to which the use of various distributed software development tools (e.g., CVS, bug tracking databases) provides a source of structure for the process.

Cognitive maps. Cognitive maps will be developed from interview data to represent and compare the mental models of the developers about the project and project team so as to gauge the degree of common knowledge and the development of shared mental models [134-137]. Metrics (e.g., number of heads, tails, domain and centrality) provided by existing software packages (e.g., Decision Explorer or CMAP2) and ad hoc developed metrics will be used to analyze and compare the different maps. In particular, the comparisons among different team members' maps will provide insights about eventual shared mental models acting within teams. We will also derive collective maps for each project. Collective maps usually represent perspectives that are common to all the members of a team. Shared perspectives derive from the comprehension of mutual positions and roles, which are fundamental to create synergies within the team.

Work plan

Based on preliminary assessment of the effort required, we are requesting funding for two graduate students. The graduate students will devote 50% effort during the academic year and 100% effort during the summers, for a total of 2200 hours/year. The graduate students will support the principal investigators in sample section, definition of constructs and variables, and will have primary responsibility for data collection and analysis, under the oversight of the PI. Two of the principal investigator will work one-third-time on the project during the summers, 1 month per year (part of this time will be supported by the PI's institution). Summers will be devoted to sample selection, interviews, design and implementation of interventions, analysis and publication of results. To support the regular schedule of data collection needed for a longitudinal study, the second PI will devote 20% of effort (1 day / week) during the academic year to data collection and analysis, as well as on-going project management and oversight. A more detailed year-by-year work plan is included as an appendix.

Conclusion

In this proposal, we develop a conceptual framework and a research plan to investigate work practices within distributed FLOSS development teams. To answer our research question, we will conduct a longitudinal in-depth study identifying and comparing the formation and evolution of distributed teams of FLOSS developers. We will study how these distributed groups develop shared mental models to guide members' behavior, roles to control access to resources, and norms and rules to shape action and the dynamics by which independent, geographically-dispersed individuals are socialized into the group.

The project will contribute to advancing knowledge and understanding of distributed teams by identifying the dynamics of distributed FLOSS teams. The study has two main strengths. First, we fill a gap in the literature with an in-depth investigation of the dynamics of developing shared mental models, role and norms and rules in FLOSS teams and of socializing new members to these structures, based on a large pool of data and a strong conceptual framework. Second, we use several different techniques to analyze the team dynamics, providing different perspectives of analysis and a more reliable portrait of what happens in the development teams. Moreover, some of data analysis techniques, such as cognitive maps and social network analysis, have not yet been used with FLOSS teams.

We expect this study to have conceptual, methodological as well as practical contributions. Understanding the dynamics of learning in a team of independent knowledge workers working in a distributed environment is important to improve the effectiveness of distributed teams and of the traditional and non-traditional organizations within which they exist. As Maier et al. suggest; "Knowledge about the process, or the know how, of learning facilitates corrections that simulate or accelerate learning" [10]. Developing a theoretical framework consolidating a number of theories to understand the dynamics within a distributed team is a contribution to the study of distributed teams and learning within organization literature [55]. Employing qualitative techniques to understand the process of learning will also be a contribution to the organizational learning methodology [138].

The project has numerous broader impacts. The study will shed light on dynamics of learning and socialization for distributed work teams, which will be valuable for leaders who intend to implement such an organizational form. Understanding the dynamics of learning and socialization can serve as guidelines (in team governance, task coordination, communication practices, mentoring, etc.) to improve performance and foster innovation. Understanding these questions is important because a network organization entails an increased use of distributed teams for a wide range of work. Distributed work teams potentially provide several benefits but the separation between members of distributed teams creates difficulties in coordination, collaboration and learning, which may ultimately result in a failure of the team to be effective [48,59,139,140]. For the potential of distributed teams to be fully realized, research is needed on the dynamics of learning and socialization. As well, findings from the study can be used to enhance the way CMC technologies are used in education or for scientific collaboration. For example, the results could be used to improve the design and facilitation of e-learning courses and distance classes. Finally, understanding FLOSS development teams may be important as they are potentially training grounds for future software developers. As Arent and Nørbjerg [141] note, in these teams, "developers collectively acquire and develop new skills and experiences".

To ensure that our study has a significant impact, we plan to broadly disseminate results through journal publications, conferences, workshops and on our Web pages. We also plan to disseminate results directly to practitioners through interactions with our advisory board and with developers, e.g., at FLOSS conferences. Our results could also potentially be incorporated into training programs. Findings about the dynamics of the learning process in FLOSS development teams can also benefit the design of technology and engineering curricula.

Proposed Work Schedule

Year	Date	Research Activities	Data Collection	Data Analysis	Product
Year One	July–September 2005	▪Assess group process for developing shared mental models, rules, norms and leadership and socializing new members in retrospective groups (case 1 & 2 in figure 3) ▪Assess changes in social structures ▪Further develop theoretical and content analytic framework	▪Interaction logs ▪Project demographics ▪Developer demographics ▪Project plans & procedures	▪Content analysis ▪Social network analysis ▪Process map ▪Statistical analysis	Initial Findings: ▪Theoretical framework ▪Content analytic framework ▪Factors that enhance or impede learning ▪Leadership characteristics suitable for distributed environments
	October–December 2005	▪Assess group process for developing shared mental models, rules, norms and leadership and socializing new members in all groups ▪Assess changes in social structures ▪Interview developers ▪Ethnography of the two newly formed groups ▪Further develop theoretical framework and content analytic framework ▪Reliability test for content analytic framework	▪Interaction logs ▪Project demographics ▪Developer demographics ▪Project plans & procedures ▪Interviews ▪Observation of developers at Apache Con (November 2005) ▪Participant Observation	▪Content analysis ▪Social network analysis ▪Process map	
	January–May 2006	▪Assess group process for developing shared mental models, rules, norms and leadership and socializing new members in all groups ▪Assess changes in social and cognitive structures ▪Interview and survey developers ▪Further develop theoretical framework and content analytic framework ▪Test reliability of content analytic framework ▪Cross analysis between groups ▪Consult with advisory board and other researcher at FLOSS conferences and academic conferences	▪Interaction logs ▪Project demographics ▪Developer demographics ▪Project plans & procedures ▪Interviews ▪Observation of developers Participant Observation ▪Survey	▪Content analysis ▪Social network analysis ▪Process map ▪Cognitive maps ▪Statistical Analysis	

Year	Date	Research Activities	Data Collection	Data Analysis	Product
Year Two	June–December 2006	▪ Assess group process for developing shared mental models, rules, norms and leadership and socializing new members in all groups ▪ Assess changes in social and cognitive structures ▪ Ethnography of the two newly formed groups ▪ Interview developers ▪ Further develop theoretical framework and content analytic framework ▪ Test reliability of content analytic framework ▪ Intervention (training for FLOSS developers in at O'Reilly Open Source Convention (June 2006) and Apache Con (Nov 2006) ▪ Consult with advisory board and other researcher at FLOSS conferences and academic conferences	▪ Interaction logs ▪ Project demographics ▪ Developer demographics ▪ Project plans & procedures ▪ Interviews ▪ Observation of developers at O'Reilly Open Source Convention (June 2006) and Apache Con (November 2006) ▪ Participant observation	▪ Content analysis ▪ Social network analysis ▪ Process map ▪ Cognitive maps	▪ Training sessions to implement interventions for two of the four groups and for ARI that center around ▪ Identified factors that enhance or impede learning ▪ Identified leadership characteristics suitable for distributed environments
	January–May 2007	▪ Assess group process for developing shared mental models, rules, norms and leadership and socializing new members in all groups ▪ Assess changes in social and cognitive structures ▪ Ethnography of the two newly formed groups ▪ Interview and survey developers ▪ Further develop theoretical framework and content analytic framework ▪ Reliability test for content analytic framework ▪ Cross analysis between groups ▪ Consult with advisory board and other researcher in FLOSS conferences and academic conferences	▪ Interaction logs ▪ Project demographics ▪ Developer demographics ▪ Project plans & procedures ▪ Interviews ▪ Observation of developers ▪ Participant observation ▪ Survey	▪ Content analysis ▪ Social network analysis ▪ Process map ▪ Cognitive maps ▪ Statistical analysis	

Year	Date	Research Activities	Data Collection	Data Analysis	Product
Year Three	June–December 2007	▪ Assess group process for developing shared mental models, rules, norms and leadership and socializing new members in all groups ▪ Assess changes in social and cognitive structures ▪ Ethnography of the two newly formed groups ▪ Interview and survey developers ▪ Further develop theoretical framework and content analytic framework ▪ Test reliability of content analytic framework ▪ Intervention (training for FLOSS developers in at O'Reilly Open Source Convention (June 2007) ▪ Consult with advisory board and other researcher in FLOSS conferences and academic conferences	▪ Interaction logs ▪ Project demographics ▪ Developer demographics ▪ Project plans & procedures ▪ Interviews ▪ Observation of developers at *O'Reilly Open Source Convention* (June 2007) and *ApacheCon* (November 2007) ▪ Participant observation	▪ Content analysis ▪ Social network analysis ▪ Process map ▪ Cognitive maps	▪ Comprehensive manuscript of findings to enhance learning and the training of leadership in distributed environment.
	January–June 2008	▪ Interview and survey developers ▪ Assess changes in social and cognitive structures ▪ Cross analysis between groups ▪ Further develop theoretical framework and content analytic framework ▪ Test reliability of content analytic framework ▪ Produce manuscripts of final findings	▪ Interaction logs ▪ Project demographics ▪ Developer demographics ▪ Project plans & procedures ▪ Interviews ▪ Surveys	▪ Content analysis ▪ Social network analysis ▪ Process map ▪ Cognitive maps ▪ Statistical Analysis	

References

1 Barley, S.R. and Tolbert, P.S. (1997) Institutionalization and structuration: Studying the links between action and institution. *Organization Studies* 18(1), 93–117

2 Garstka, J.J. (2000) Network Centric Warfare: An Overview of Emerging Theory. *PHALANX: Journal of the Military Operations Research Society* 33(4), 1–33

3 Cebrowski, A.K. and Garstka, J.J. (1998) Network-centric warfare: Its origin and future. In *U.S. Naval Institute Proceedings* (Vol. 124), pp. 28–35

4 Stamelos, I., Angelis, L., Oikonomou, A. and Bleris, G.L. (2002) Code quality analysis in open source software development. *Information Systems Journal* 12(1), 43–60

5 Raymond, E.S. (1998) The cathedral and the bazaar. *First Monday* 3(3)

6 Wayner, P. (2000) *Free For All*, HarperCollins

7 Herbsleb, J.D. and Grinter, R.E. (1999) Splitting the organization and integrating the code: Conway's law revisited. In *Proceedings of the International Conference on Software Engineering (ICSE '99)*, pp. 85–95, Los Angeles, CA: ACM

8 Glance, D.G. (2004) Release criteria for the Linux kernel. *First Monday* 9(4)

9 Scacchi, W. (2002) Understanding the requirements for developing Open Source Software systems. *IEE Proceedings Software* 149(1), 24–39

10 Maier, G.W., Prange, C. and Rosenstiel, L. (2001) Psychological perspectives on organizational learning. In *Handbook of Organizational Learning and Knowledge* (Dierkes, M., Berthoin Antal, A., Child, J. and Nonaka, I., eds.), pp. 14–34, New York: Oxford Press

11 Huber, G.P. (1991) Organizational learning: The contributing processes and the literatures. *Organization Science* 2(1), 88–115

12 Di Bona, C., Ockman, S. and Stone, M., eds (1999) *Open Sources: Voices from the Open Source Revolution*, O'Reilly & Associates

13 Kogut, B. and Metiu, A. (2001) Open-source software development and distributed innovation. *Oxford Review of Economic Policy* 17(2), 248–264

14 Lerner, J. and Tirole, J. (2001) The open source movement: Key research questions. *European Economic Review* 45, 819–826

15 Hertel, G., Niedner, S. and Herrmann, S. (n.d.) *Motivation of Software Developers in Open Source Projects: An Internet-based Survey of Contributors to the Linux Kernel*, University of Kiel

16 Hann, I.-H., Roberts, J., Slaughter, S. and Fielding, R. (2002) Economic incentives for participating in open source software projects. In *Proceedings of the Twenty-Third International Conference on Information Systems*, pp. 365–372

17 Bessen, J. (2002) *Open Source Software: Free Provision of Complex Public Goods*, Research on Innovation

18 Franck, E. and Jungwirth, C. (2002) *Reconciling investors and donators: The governance structure of open source*, Working Paper (No. 8), Lehrstuhl für Unternehmensführung und -politik, Universität Zürich

19 Markus, M.L., Manville, B. and Agres, E.C. (2000) What makes a virtual organization work? *Sloan Management Review* 42(1), 13–26

20 Ljungberg, J. (2000) Open Source Movements as a Model for Organizing. *European Journal of Information Systems* 9(4)

21 Stewart, K.J. and Ammeter, T. (2002) An exploratory study of factors influencing the level of vitality and popularity of open source projects. In *Proceedings of the Twenty-Third International Conference on Information Systems*, pp. 853–857

22 Bezroukov, N. (1999) Open source software development as a special type of academic research (critique of vulgar raymondism). *First Monday* 4(10)
23 Bezroukov, N. (1999) A second look at the Cathedral and the Bazaar. *First Monday* 4(12)
24 Guzzo, R.A. and Dickson, M.W. (1996) Teams in organizations: Recent research on performance effectiveness. *Annual Review of Psychology* 47, 307–338
25 O'Leary, M., Orlikowski, W.J. and Yates, J. (2002) Distributed work over the centuries: Trust and control in the Hudson's Bay Company, 1670–1826. In *Distributed Work* (Hinds, P. and Kiesler, S., eds.), pp. 27–54, Cambridge, MA: MIT Press
26 Ahuja, M.K., Carley, K. and Galletta, D.F. (1997) Individual performance in distributed design groups: An empirical study. In *SIGCPR Conference*, pp. 160–170, ACM, San Francisco
27 Nejmeh, B.A. (1994) Internet: A strategic tool for the software enterprise. *Communications of the ACM* 37(11), 23–27
28 Scacchi, W. (1991) The software infrastructure for a distributed software factory. *Software Engineering Journal* 6(5), 355–369
29 Watson-Manheim, M.B., Chudoba, K.M. and Crowston, K. (2002) Discontinuities and continuities: A new way to understand virtual work. *Information, Technology and People* 15(3), 191–209
30 van Fenema, P.C. (2002) Coordination and control of globally distributed software projects. In *Erasmus Research Institute of Management*, pp. 572, Erasmus University
31 Armstrong, D.J. and Cole, P. (2002) Managing distance and differences in geographically distributed work groups. In *Distributed Work* (Hinds, P. and Kiesler, S., eds.), pp. 167–186, Cambridge, MA: MIT Press
32 Curtis, B., Walz, D. and Elam, J.J. (1990) Studying the process of software design teams. In *Proceedings of the 5th international software process workshop on experience with software process models*, pp. 52–53, Kennebunkport, Maine, United States
33 Espinosa, J.A., Kraut, R.E., Lerch, J.F., Slaughter, S.A., Herbsleb, J.D. and Mockus, A. (2001) Shared mental models and coordination in large-scale, distributed software development. In *Twenty-Second International Conference on Information Systems*, pp. 513–518, New Orleans, LA
34 Curtis, B., Krasner, H. and Iscoe, N. (1988) A field study of the software design process for large systems. *CACM* 31(11), 1268–1287
35 Walz, D.B., Elam, J.J. and Curtis, B. (1993) Inside a software design team: knowledge acquisition, sharing, and integration. *Communications of the ACM* 36(10), 63–77
36 Humphrey, W.S. (2000) *Introduction to team software process*, Addison-Wesley
37 Sawyer, S. and Guinan, P.J. (1998) Software development: Processes and performance. *IBM Systems Journal* 37(4), 552–568
38 Brooks, F.P., Jr. (1975) *The Mythical Man-month: Essays on Software Engineering*, Addison-Wesley
39 Seaman, C.B. and Basili, V.R. (1997) *Communication and Organization in Software Development: An Empirical Study*, Institute for Advanced Computer Studies, University of Maryland
40 Ocker, R.J. and Fjermestad, J. (2000) High versus low performing virtual design teams: A preliminary analysis of communication. In *Proceedings of the 33rd Hawaii International Conference on System Sciences*, pp. 10 pages
41 Mockus, A., Fielding, R.T. and Herbsleb, J.D. (2000) A case study of Open Source Soft-

ware development: The Apache server. In *Proceedings of ICSE'2000*, pp. 11 pages

42 Herbsleb, J.D., Mockus, A., Finholt, T.A. and Grinter, R.E. (2001) An empirical study of global software development: Distance and speed. In *Proceedings of the International Conference on Software Engineering (ICSE 2001)*, pp. 81–90, Toronto, Canada:

43 Bandow, D. (1997) Geographically distributed work groups and IT: A case study of working relationships and IS professionals. In *Proceedings of the SIGCPR Conference*, pp. 87–92

44 Mark, G. (2002) Conventions for coordinating electronic distributed work: A longitudinal study of groupware use. In *Distributed Work* (Hinds, P. and Kiesler, S., eds.), pp. 259–282, Cambridge, MA: MIT Press

45 Cascio, W.F. and Shurygailo, S. (2003) E-Leadership and virtual teams. *Organizational Dynamics* 31(4), 362–376

46 Zigurs, I. (2003) Leadership in virtual teams: Oxymoron or opportunity? *Organizational Dynamics* 31(4), 339-351

47 Jarvenpaa, S.L., Knoll, K. and Leidner, D.E. (1998) Is Anybody Out There? Antecedents of Trust in Global Virtual Teams. *Journal of Information Systems* 14(4), 29–64

48 Jarvenpaa, S.L. and Leidner, D.E. (1999) Communication and trust in global virtual teams. *Organization Science* 10(6), 791–815

49 Kayworth, T.R. and Leidner, D.E. (2002) Leadership effectiveness in global virtual teams. *Journal of Management Information Systems* 18(3), 7–40

50 Tyran, K.L., Tyran, C.K. and Shepherd, M. (2003) Exploring emergent leadership in virtual teams. In *Virtual Teams That Work: Creating Conditions for Virtual Team Effectiveness* (Gibbon, C.B. and Cohen, S.G., eds.), pp. 183–195, San Francisco: Jossey-Bass

51 Yoo, Y. and Alavi, M. (2002) *Electronic Mail Usage Pattern of Emergent Leaders in Distributed Teams*, Available from http://weatherhead.cwru.edu/sprouts/2002/020309.pdf

52 Lerner, J. and Tirole, J. (2000) *The Simple Economics of Open Source* (NBER Working Paper w7600), The National Bureau of Economic Research, Inc., http://papers.nber.org/papers/W7600

53 Edwards, K. (2000) When beggars become choosers. *First Monday* 5(10)

54 Orlikowski, W.J. (2002) Knowing in practice: Enacting a collective capability in distributed organizing. *Organization Science* 13(3), 249–273

55 Robey, D., Khoo, H.M. and Powers, C. (2000) Situated-learning in cross-functional virtual teams. *IEEE Transactions on Professional Communication*(Feb/Mar), 51–66

56 Butler, B., Sproull, L., Kiesler, S. and Kraut, R. (2002) Community effort in online groups: Who does the work and why? In *Leadership at a Distance* (Weisband, S. and Atwater, L., eds.)

57 Grabowski, M. and Roberts, K.H. (1999) Risk mitigation in virtual organizations. *Organization Science* 10(6), 704–721

58 Herbsleb, J.D. and Grinter, R.E. (1999) Architectures, coordination, and distance: Conway's law and beyond. *IEEE Software*(September/October), 63–70

59 Kraut, R.E., Steinfield, C., Chan, A.P., Butler, B. and Hoag, A. (1999) Coordination and virtualization: The role of electronic networks and personal relationships. *Organization Science* 10(6), 722–740

60 Kiesler, S. and Cummings, J. (2002) What do we know about proximity and distance in work groups? A legacy of research. In *Distributed Work* (Hinds, P. and Kiesler, S., eds.), pp. 57–80, Cambridge, MA: MIT Press

61 Stogdill, R.M. (1974) *Handbook of leadership: A survey of theory and research*, Free Press
62 Blake, R.R. and Mouton, J.S. (1964) *The Managerial Grid*, Gulf
63 Fiedler, F.E. (1978) The contingency model and the dynamics of the leadership process. In *Advances in experimental social psychology* (Vol. 11) (Berkowitz, L., ed.), pp. 59–112, New York: Academic
64 Hersey, P. and Blanchard, K.H. (1988) *Management of Organizational Behavior: Utilizing Human Resources*, Prentice Hall
65 Pavitt, C. (1998) *Small Group Communication: A Theoretical Approach (3rd Ed)*, Available from http://www.udel.edu/communication/pavitt/bookindex.htm, accessed 30 April 2004
66 Bales, R.F. (1953) The equilibrium problem in small groups. In *Working papers in the theory of action* (Parsons, T., Bales, R.F. and Shils, E.A., eds.), pp. 111–161, Glencoe, IL: Free Press
67 Ancona, D.G. and Caldwell, D.F. (1988) Beyond task and maintenance: Defining external functions in groups. *Group and Organization Studies* 13, 468–494
68 Grant, R.M. (1996) Toward a knowledge-based theory of the firm. *Strategic Management Journal* 17(Winter), 109–122
69 Stein, R.T. and Heller, T. (1983) The relationship of participation rates to leadership status: A meta-analysis. In *Small Groups and Social Interaction* (Blumberg, H.H., Hare, A.P., Kent, V. and Davies, M., eds.), pp. 401–406, Chichester, UK: John Wiley & Sons
70 Mullen, B., Salas, E. and Driskell, J.E. (1989) Salience, motivation, and artifact as contributions to the relation between participation rate and leadership. *Journal of Experimental Social Psychology* 25, 545–559
71 Bales, R.F. and Slater, P.E. (1955) Role differentiation in small decision-making groups. In *Family, socialization and interaction process* (Parsons, T. and Bales, R.F., eds.), pp. 259–306, Glencoe, IL: Free Press
72 Baker, D.C. (1990) A qualitative and quantitative analysis of verbal style and the elimination of potential leaders in small groups. *Communication Quarterly* 38, 13–26
73 Ketrow, S.M. (1991) Communication role specializations and perceptions of leadership. *Small Group Research* 22, 492–514
74 Zigurs, I. and Kozar, K. (1994) An exploratory study of roles in computer-supported groups. *MIS Quarterly* 18(3), 277–297
75 Heckman, R., Maswick, D., Rodgers, J., Ruthen, K. and Wee, G. (2000) The impact of information technology on roles and role processes in small groups. In *Case Studies on Information Technology in Higher Education: Implications for Policy and Practice* (Petrides, L.A., ed.), pp. 157–167, Hershey, PA: Idea Group Publishing
76 Garvin, D.A. (1991) Barriers and gateways to learning. In *Education for Judgement: The Art of Discussion Leadership* (Christensen, C.R., Garvin, D.A. & Sweet, A., ed.), pp. 3–14, Boston: Harvard Business School Press
77 Argyris, C. (1999) *On Organizational Learning*, Blackwell
78 Senge, P.M. (1990) *The Fifth Discipline: The Art and Practice of the Learning Organization*, Century Business
79 Lin, F. and Lin, S. (2001) A Conceptual Model for Virtual Organizational Learning. *Journal of Organizational Computing and Electronic Commerce* 11(3), 155–178
80 Swieringa, J. and Wierdsma, A. (1992) *Becoming a Learning Organization*, Addison-Wesley
81 Hayes, J. and Allinson, C.W. (1998) Cognitive style and the theory and practice of individ-

ual and collective learning in organizations. *Human Relations* 51(7), 847-871

82 March, J.G., Schulz, M. and Zhou, X. (2000) *The Dynamics of Rules: Change in Written Organizational Codes*, Stanford University Press

83 Urch Druskat, V. and Kayes, D.C. (2000) Learning versus performance in short-term project teams. *Small Group Research* 31(3), 328–353

84 Barley, S.R. (1986) Technology as an occasion for structuring: Evidence from the observation of CT scanners and the social order of radiology departments. *Administrative Sciences Quarterly* 31, 78–109

85 Orlikowski, W.J. (1992) The duality of technology: Rethinking the concept of technology in organizations. *Organization Science* 3(3), 398–427

86 DeSanctis, G. and Poole, M.S. (1994) Capturing the complexity in advanced technology use: Adaptive structuration theory. *Organization Science* 5(2), 121–147

87 Walsham, G. (1993) *Interpreting Information Systems in Organizations*, John-Wiley

88 Newman, M. and Robey, D. (1992) A social process model of user-analyst relationships. *MIS Quarterly* 16(2), 249–266

89 Giddens, A. (1984) *The Constitution of Society: Outline of the Theory of Structuration*, University of California

90 Sarason, Y. (1995) A model of organizational transformation: The incorporation of organizational identity into a structuration theory framework. *Academy of Management Journal*(Best papers proceedings), 47–51

91 Gregory, D. (1989) Presences and absences: Time-space relations and structuration theory. In *Social Theory of Modern Societies: Anthony Giddens and His Critics*, Cambridge: Cambridge University Press

92 Cassell, P., ed. (1993) *The Giddens Reader*, Stanford University Press

93 Orlikowski, W.J. and Yates, J. (1994) Genre repertoire: The structuring of communicative practices in organizations. *Administrative Sciences Quarterly* 33, 541–574

94 Hackman, J.R. (1986) The design of work teams. In *The Handbook of Organizational Behavior* (Lorsch, J.W., ed.), pp. 315–342, Englewood Cliffs, NJ: Prentice-Hall

95 Finholt, T. and Sproull, L.S. (1990) Electronic groups at work. *Organization Science* 1(1), 41–64

96 Stein, E.W. and Vandenbosch, B. (1996) Organizational learning during advanced system development: Opportunities and obstacles. *Journal of Management Information Systems* 13(2), 115–136

97 Cannon-Bowers, J.A. and Salas, E. (1993) Shared mental models in expert decision making. In *Individual and Group Decision Making* (Castellan, N.J., ed.), pp. 221-246, Hillsdale, NJ: Lawrence Erlbaum Associates

98 Dougherty, D. (1992) Interpretive barriers to successful product innovation in large firms. *Organization Science* 3(2), 179–202

99 Levesque, L.L., Wilson, J.M. and Wholey, D.R. (2001) Cognitive divergence and shared mental models in software development project teams. *Journal of Organization Behavior* 22, 135–144

100 Walton, R.E. and Hackman, J.R. (1986) Groups under contrasting management strategies. In *Designing Effective Work Groups* (Goodman, P.S. and Associates, eds.), pp. 168–201, San Francisco, CA: Jossey-Bass

101 Brown, J.S. and Duguid, P. (1991) Organizational learning and communities-of-practice: Toward a unified view of working, learning, and innovation. *Organization Science* 2(1), 40–57

102 Mohammed, S. and Dumville, B.C. (2001) Team mental models in a team knowledge framework: Expanding theory and measurement across disciplinary boundaries. *Journal of Organizational Behavior* 22(2), 89–106

103 Rentsch, J.R. and Klimonski, R.J. (2001) Why do 'great minds' think alike?: Antecedents of team member schema agreement. *Journal of Organizational Behavior* 22(2), 107–120

104 Moon, J.Y. and Sproull, L. (2000) Essence of distributed work: The case of Linux kernel. *First Monday* 5(11)

105 Cox, A. (1998) *Cathedrals, Bazaars and the Town Council*, Available from http://slashdot.org/features/98/10/13/1423253.shtml, accessed 22 March 2004

106 Gacek, C., Lawrie, T. and Arief, B. (n.d.) *The many meanings of Open Source*, Unpublished manuscript, Centre for Software Reliability, Department of Computing Science, University of Newcastle

107 Fielding, R.T. (1997) *The Apache Group: A case study of Internet collaboration and virtual communities*, Available from http://www.ics.uci.edu/fielding/talks/ssapache/overview.htm.

108 Hecker, F. (1999) *Mozilla at one: A look back and ahead*, Available from http://www.mozilla.org/mozilla-at-one.html

109 Cubranic, D. and Booth, K.S. (1999) Coordinating Open Source Software development. In *Proceedings of the 7th IEEE Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises*

110 Argyris, C. and Schön, D.A. (1978) *Organizational Learning*, Addison-Wesley

111 Susman, G.I. and Evered, R.D. (1978) An assessment of the scientific merits of action research. *Administrative Science Quarterly* 23, 582–603

112 Rapoport, R.N. (1970) Three dilemmas in action research—With special reference to the Tavistock experiences. *Human Relations* 23(6), 499–513

113 Aguinis, H. (1993) Action research and scientific method: Presumed discrepancies and actual similarities. *Journal of Applied Behavioural Science* 29(4), 416–431

114 Eisenhardt, K.M. (1991) Better stories and better constructs: The case for rigor and comparative logic. *Academy of Management Review* 16(3), 620–627

115 Yin, R.K. (1984) *Case study research: Design and methods*, Sage

116 Krishnamurthy, S. (2002) *Cave or Community? An Empirical Examination of 100 Mature Open Source Projects*, University of Washington, Bothell

117 Hare, A.P. (1976) *Handbook of Small Group Research*, Free Press

118 Baskerville, R.L. and Wood-Harper, A.T. (1996) A critical perspective on action research as a method for information systems research. *Journal of Information Technology* 11, 235–246

119 Lee, G.K. and Cole, R.E. (2000) *The Linux Kernel Development As A Model of Open Source Knowledge Creation*, Unpublished manuscript, Haas School of Business, University of California, Berkeley

120 Webb, E. and Weick, K.E. (1979) Unobtrusive measures in organizational theory: A reminder. *Administrative Science Quarterly* 24(4), 650–659

121 Nardi, B.A. and Whittaker, S. (2002) The place of face-to-face communication in distributed work. In *Distributed Work* (Hinds, P. and Kiesler, S., eds.), pp. 83–110, Cambridge, MA: MIT Press

122 Edwards, K. (2001) Epistemic communities, situated learning and Open Source Software development. In *Epistemic Cultures and the Practice of Interdisciplinarity Workshop*, NTNU, Trondheim

123 Mortensen, M. and Hinds, P. (2002) Fuzzy teams: Boundary disagreement in distributed and collocated teams. In *Distributed Work* (Hinds, P. and Kiesler, S., eds.), pp. 284–308, Cambridge, MA: MIT Press

124 Weisband, S. (2002) Maintaining awareness in distributed team collaboration: Implications for leadership and performance. In *Distributed Work* (Hinds, P. and Kiesler, S., eds.), pp. 311–333, Cambridge, MA: MIT Press

125 Miles, M.B. and Huberman, A.M. (1994) *Qualitative Data Analysis : An Expanded Sourcebook*, Sage Publications

126 Edmondson, A. (1999) Psychological safety and learning behavior in work teams. *Administrative Science Quarterly* 44(2), 350-383

127 Heckman, R. and Annabi, H. (2003) A content analytic comparison of FTF and ALN case-study discussions. In *36th Annual Hawaii International Conference on System Sciences (HICSS'03)*, IEEE Press, Big Island, Hawaii

128 Baker-Brown, G., Ballard, E., Bluck, S., DeVries, B., Suedfeld, P. and Tetlock, P. (1990) *Coding Manual for Conceptual/Integrative Complexity*, University of British Columbia and University of California, Berkeley

129 Madey, G., Freeh, V. and Tynan, R. (2002) The Open Source Software development phenomenon: An analysis based on social network theory. In *Proceedings of the Eighth Americas Conference on Information Systems*, pp. 1806–1815

130 van de Ven, A.H. and Poole, M.S. (1990) Methods for studying innovation development in the Minnesota Innovations Research Program. *Organization Science* 1(3), 313–335

131 Yamauchi, Y., Yokozawa, M., Shinohara, T. and Ishida, T. (2000) Collaboration with lean media: How open-source software succeeds. In *Proceedings of CSCW'00*, pp. 329–338, Philadelphia, PA:

132 Abbott, A. (1990) A primer on sequence methods. *Organization Science* 1(4), 375–392

133 Crowston, K. and Osborn, C.S. (2003) A coordination theory approach to process description and redesign. In *Organizing Business Knowledge: The MIT Process Handbook* (Malone, T.W., Crowston, K. and Herman, G., eds.), Cambridge, MA: MIT Press

134 Carley, K.M. and Palmquist, M. (1992) Extracting, representing and analyzing mental models. *Social Forces* 70(3), 601–636

135 Carley, K.M. (1997) Extracting team mental models through textual analysis. *Journal of Organizational Behavior* 18, 533–558

136 Langfield-Smith, K. (1992) Exploring the need for a shared cognitive map. *Journal of management studies* 29(3), 349-368

137 Nadkarni, S. and Nah, F.F.-H. (2003) Aggregated causal maps: An approach to elicit and aggregate the knowledge of multiple experts. *Communications of the Association for Information Systems* 12, 406–436

138 Miner, A.S. and Mezias, S.J. (1996) Ugly Duckling No More: Pasts and Futures of Organizational learning. *Organization Science* 7(1), 88–99

139 Bélanger, F. and Collins, R. (1998) Distributed Work Arrangements: A Research Framework. *The Information Society* 14(2), 137–152

140 Carmel, E. and Agarwal, R. (2001) Tactical approaches for alleviating distance in global software development. *IEEE Software*(March/April), 22–29

141 Arent, J. and Nørbjerg, J. (2000) Software Process Improvement as Organizational Knowledge Creation: A Multiple Case Analysis. In *Proceedings of the 33rd Hawaii International Conference on System Sciences*, pp. 11 pages, IEEE Press