EMERGENT DECISION-MAKING PRACTICES IN FREE/LIBRE OPEN SOURCE SOFTWARE (FLOSS) DEVELOPMENT TEAMS

ROBERT HECKMAN, KEVIN CROWSTON, U. YELIZ ESERYEL, JAMES HOWISON, EILEEN ALLEN, QING LI

School of Information Studies,
Syracuse University 344 Hinds Hall Syracuse, NY 13244-4100 USA
WWW home page: http://floss.syr.edu
{rheckman, crowston, uyeserye, jhowison, eeallen, qli03}@syr.edu

Abstract: We seek to identify work practices that make Free/Libre Open Source Software (FLOSS) development teams effective. Particularly important to team effectiveness is decision making. In this paper, we report on an inductive qualitative analysis of 360 decision episodes of six FLOSS development teams. Our analysis revealed diversity in decision-making practices that seem to be related to differences in overall team characteristics and effectiveness.

Key words: Decision making practices; free/libre open source software development teams; team effectiveness; FLOSS; OSS

1. 1 INTRODUCTION

In Free/Libre Open Source Software (FLOSS) teams, decision-making practices emerge from the interactions of the team members rather than from organizational context. Discontinuities among team members make such emergence and indeed any kind of consistent decision process harder to attain, yet effective teams seem to have developed productive ways of making decisions. Developers contribute from around the world, meet face-

to-face infrequently (some not at all), and coordinate their activity primarily by means of information communication technologies (ICT) (Raymond, 1998a; Wayner, 2000). Since FLOSS teams are representative of self-organizing teams, because they have shared goals and a user base and members to satisfy, and are interdependent in terms of tasks and roles, our findings will have broader implications for understanding other technology supported self-organizing teams.

Our objectives for this paper are two-fold: First, to present a descriptive analysis of the range and evolution of decision-making practices of FLOSS teams based on longitudinal observation of 120 decision episodes that took place in 6 naturally occurring teams. We chose projects that are similar in market size potential and software development stage, that use similar tools, and belong to one of two software categories: Instant Messaging (IM) Clients and Enterprise Resource Planning (ERP) Systems. We present this description in the form of multiple case studies that compare and contrast decision-making practices between teams.

A second objective is to relate differences in team work practices to team effectiveness. Because we compare teams that differ in effectiveness but are similar in other ways, we provide suggestions for future research on the relationship between decision-making practices and team effectiveness. This comparison, between teams in two different software categories, also enables us to understand how software properties such as software complexity, target market, and team nature can affect the decision making process.

2. LITERATURE REVIEW

In this section, we briefly review literature relevant to our study of decision-making practices in FLOSS teams.

Dean and Sharfman (1996) suggest a close link between decision making processes and decision effectiveness. Guzzo and Salas (1995) suggest a close tie between effective decision making and overall team effectiveness and the importance of understanding the practices by which decisions are actually made in teams. In the information systems (IS) literature more particularly, there have been numerous studies of ICT support for group decision making (e.g., DeSanctis & Gallupe, 1987a; Fjermestad & Hiltz, 1998/1999; Turoff, Hiltz, Bahgat, & Rana, 1993). Huber et al. (1986) suggest that decisions that groups need to make are becoming increasingly more complex, and requires greater participations and a faster decision making process. DeSanctis and Gallupe (1987b) expect greater and more even participation to yield desirable effects for the group. High participation from a group allows pooling of more

resources and promotes error checking, thus enabling better decisions (DeSanctis & Gallupe, 1987b; Hackman & Kaplan, 1974; Holloman & Hendrick, 1972). Small group research suggests that higher participation in group decisions increases the acceptance of these decisions and members' increased sense of responsibility for those decisions (Bedau, 1984; DeSanctis & Gallupe, 1987b; Hackman & Kaplan, 1974). The small group research literature also identifies that higher participation in group decisions increases the level of group cohesion and individuals' satisfaction with the group (e.g., Hare, 1976).

High participation in group decisions may potentially slow down the decision making process, resulting in dissatisfaction with the decision making process. Yet, slowing processes in the FLOSS environment may be less problematic given that FLOSS projects are protected from the type of competition that their for-profit counterparts face.

Many of the studies on decision making in the IS field have been design focused, offering important suggestions to improve the process and quality of team decisions. Studies of groups in action have tended to adopt experimental methods and focus on single episodes of decision making rather than on practices over the life of an intact team (though there are exceptions, such as (Eden & Ackermann, 2001)). Broadly speaking, there are few studies that examine the kinds of decision processes that emerge in intact self-organizing teams, how these practices evolve over time, and how they contribute to overall team effectiveness. These decision processes include, but aren't limited to, how the decision process gets initiated and concluded, the types and roles of participants, and the frequency and quality of participation.

One common concern in several studies of FLOSS teams' decision making, has been the style of participation. At one extreme is a style where decisions are primarily made by a few central participants, even a single individual, as in Linux, where Linus Torvalds originally made most of the decisions for the team (Moon & Sproull, 2000). Such a decision style has been characterized as a "benevolent dictatorship" (Raymond, 1998b). On the other extreme are teams with a decentralized communications structure and more consultative decision-making style. Some teams even settle decisions by voting (Fielding, 1999). Although participation in decision making by a few key people at the core versus the people at all levels has been described in these studies, the connection between participation style and team effectiveness isn't clear. In addition, the participation in decision making might evolve over time as the project evolves. Fitzgerald (2006) suggests that a small group will control decision making early in the life of a project, but as the project grows, more developers will be involved. German (2003)

documents such a transition in the case of the Gnome project. Thus, not only the extent and frequency of participation, but also the evolution of decision participation over time may influence the relationship between decision making practices and team effectiveness.

3. METHOD

To analyze decision-making practices in open-source projects, our research employs a multiple case study methodology, focused primarily on content analysis of decision-making discussions. To find these discussions, we analyzed the email discourse between administrators, developers, and users that takes place on the developers' e-mail lists or forums, which are the primary communication venue for the teams. Archives of these lists are available on project websites and from repositories such as SourceForge.net[1].

3.1 Case selection

We chose six FLOSS projects by considering several dimensions to balance maximization of variability and control of unwanted systematic variance. First, we controlled for topic. Projects within a single topic category are potential competitors, making comparisons of outcomes such as downloads between these projects valid. On the other hand, we wanted to have projects at different levels of complexity to provide for variability. Accordingly we picked three projects that develop Enterprise Resource Planning (ERP) systems (Compiere, WebERP and Apache OFBiz) and three teams that develop Instant Messenger (IM) clients (Gaim, aMSN and Fire). ERP projects are more complex than IM projects since they have high software code interdependencies, and many external constraints such as accounting rules and legal reporting requirements. One, Compiere, originated as a closed-source project, offering an opportunity to examine the consequences of that history.

Second, to minimize unwanted variance, we chose projects that are roughly similar in age and status (production/stable.) Projects at this stage have relatively developed membership and sufficient team history, yet the software code still has room for improvement, which enables us to observe rich team interaction processes. Third, the projects we chose varied in

[1] Because postings to lists are intended to be publicly accessible, our human subjects review board considers them public behavior, and so does not require formal consent to study them.

effectiveness. Project effectiveness is a multi-dimensional construct, including success of the project's outputs, team member satisfaction and continued project activities (Hackman, 1987). We therefore applied the multivariate approach to effectiveness in the FLOSS context suggested by Crowston et al. (2006) aiming to discover a rank order within the IM and

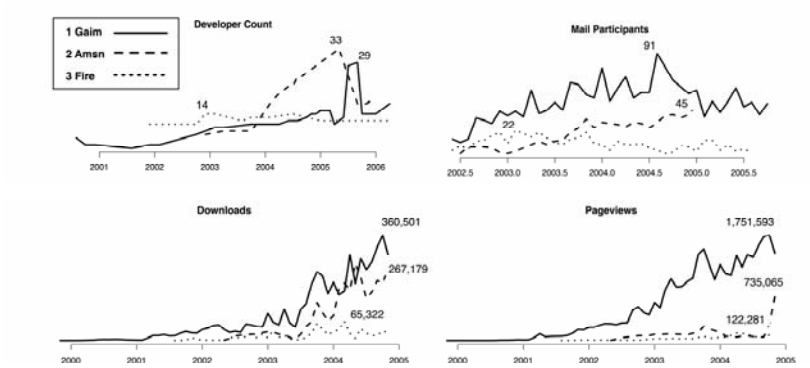

Fig. 1. Comparison of Effectiveness Measures for IM Projects

ERP categories. Project outputs were measured by downloads and page views, developer satisfaction was measured through development numbers and participation on the developer mailing lists. The array of measures presented in Figures 1 and 2 use data collected by the FLOSSmole project (Howison, Conclin, & Crowston, 2006) from the project establishment in SourceForge until around March 2006. According to analyses shown in Figures 1 and 2, the most effective IM project is Gaim, followed by aMSN then Fire, and the most effective ERP project is Compiere followed by OFBiz then WebERP.

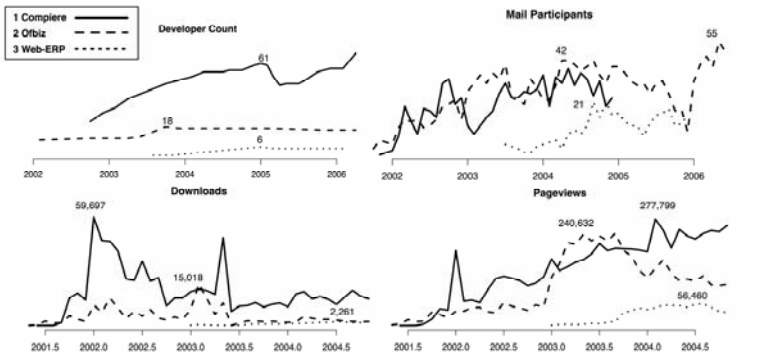

Figure 2. Comparison of Effectiveness Measures for ERP Projects

3.2 Unit of Analysis: Decision Episodes

We selected the decision episode as our primary unit of analysis. We define a decision episode as a sequence of e-mail messages that begins with a triggering message presenting an opportunity for choice (such as a feature request or a report of a software bug), includes discussion related to the issue, and an announcement of a decision about the opportunity.

We differentiated between decision episodes that focus on software code-related day-to-day decisions and those that focus on long-term strategic decisions (such as membership, infrastructure and marketing decisions). In keeping with our desire to focus on likely similarities to other forms of distributed teams, this paper focuses on the software code-related episodes.

In order to observe potential changes in decision-making processes and norms over time, we sampled 20 decision episodes from three comparable time periods in each project's life. For each project, the beginning and the ending periods are the first and last 20 decision episodes observable on the developer mailing list by May 2006. The middle period for each project consisted of 20 episodes surrounding a major software release approximately halfway between the beginning and ending periods. Figure 3 shows the specific time periods sampled for each project. Note that the sample periods differ in length due to different rates of development in the projects.

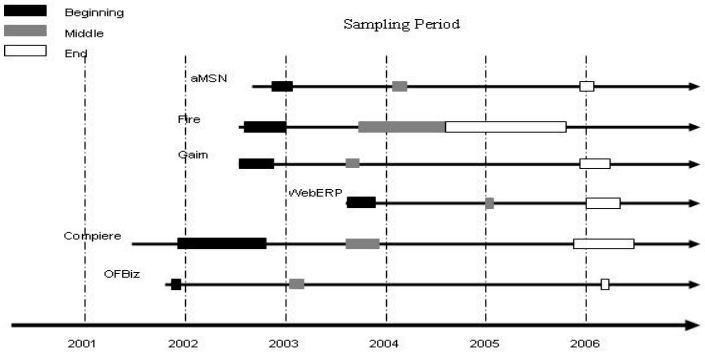

Fig. 3. Sampling Periods of IM and ERP Projects

3.3 Analysis and Coding of Episodes

We began analysis of decision episodes by coding observable, manifest elements of content that are directly related to the decision-making typologies described in the literature above. We coded: number of messages

per episode, duration of the episode (in days), total number of participants in the episode, and the role of each message's sender: project administrator, developer (if listed developer according to the project webpage on SourceForge) or non-developer (if not listed on SourceForge). We considered administrators and developers "core" members of the team, and non-developers "peripheral" members.

Subsequent coding was inductive, with three independent analysts reading the episodes in order to understand the salient features of the decision process. Through several iterations, at least two independent coders identified and agreed upon four additional latent variables that were important to the decision process. Each episode was then formally coded by at least two analysts, with all disagreements discussed and reconciled to achieve essentially complete agreement. The latent variables identified and coded are decision trigger type, decision process complexity, decision announcement, and decision type.

3.3.1 Decision Trigger Type

One goal of our inductive content analysis was to understand the types of triggers that presented decision opportunities for the group. Thus, we developed a typology of triggers. Decision episodes about code are triggered by: (1) bug reports, (2) feature requests, (3) problem reports, which are different from bug reports since they may also include problems that end-users may be facing due to hardware, software or process reasons, (4) patch submissions (5) to-do lists and (6) mixed triggers that include one or more different trigger types.

3.3.2 Decision Process Complexity

Inductive analysis also indicated that some episodes required more complex decision paths than others. For example, some episodes involve a single choice that responds to a single straightforward trigger. These were coded as "Single." Others responded in a linear, straightforward fashion to a trigger that contained multiple opportunities for choice (e.g., a release to-do list). These were coded as "Multiple-Simple." The most complex episodes were not straightforward or linear in nature. Regardless of the nature of the initial trigger, in these episodes new, sometimes unrelated triggers created additional opportunities for choice. The initial problem(s) might be solved or not, and the new problems introduced might also remain unsolved. These episodes, coded "Multiple-Complex," closely resemble the garbage can decision opportunities described by Cohen et al. (1972) in that a

straightforward, sequential “problem-resolution” decision process was not observed.

3.3.3 Decision Announcement

In order to reliably determine that a decision had truly been reached, our independent coders coded the statement(s) that confirmed that a decision had been reached.

3.3.4 Decision Type

Our analysis also coded when the main decision announcements reflected either acceptance or rejection of a need for code change, acceptance or rejection of the suggested code change, or both.

This initial typology of latent variables provides the ability to concisely describe multiple characteristics of the decision making process, and allows us to measure the participation of various members in decision making, thus contributing to our first objective, providing a rich description of the evolution of decision-making practices over time and the connection between decision making and FLOSS effectiveness.

4. FINDINGS

Our research objectives were to present a descriptive analysis of the range and evolution of decision-making participation in FLOSS teams, and to relate differences in these work practices to team effectiveness. In order to do that we present below differences in decision episode participation between more and less effective teams. We begin by first discussing overall participation in decision episodes. We then discuss relative participation by core and peripheral members, and finally, present an analysis of who triggers decision episodes and who announces decisions.

4.1 Participation in Decision Episodes

The overall number of participants in episodes increases from the first to last period for effective projects such as Gaim, aMSN and OFBiz (see Figure 4). Fire, the least-effective IM project, did not see an increase in the average number of participants per episode. Similarly, WebERP, the least effective of the 3 ERP projects, increased its average number of participants in the middle period, but did not maintain the participation. Compiere, despite its

apparently high effectiveness, exhibits a different participation behavior than the other projects, perhaps due to a project management style that remains from its closed-source days.

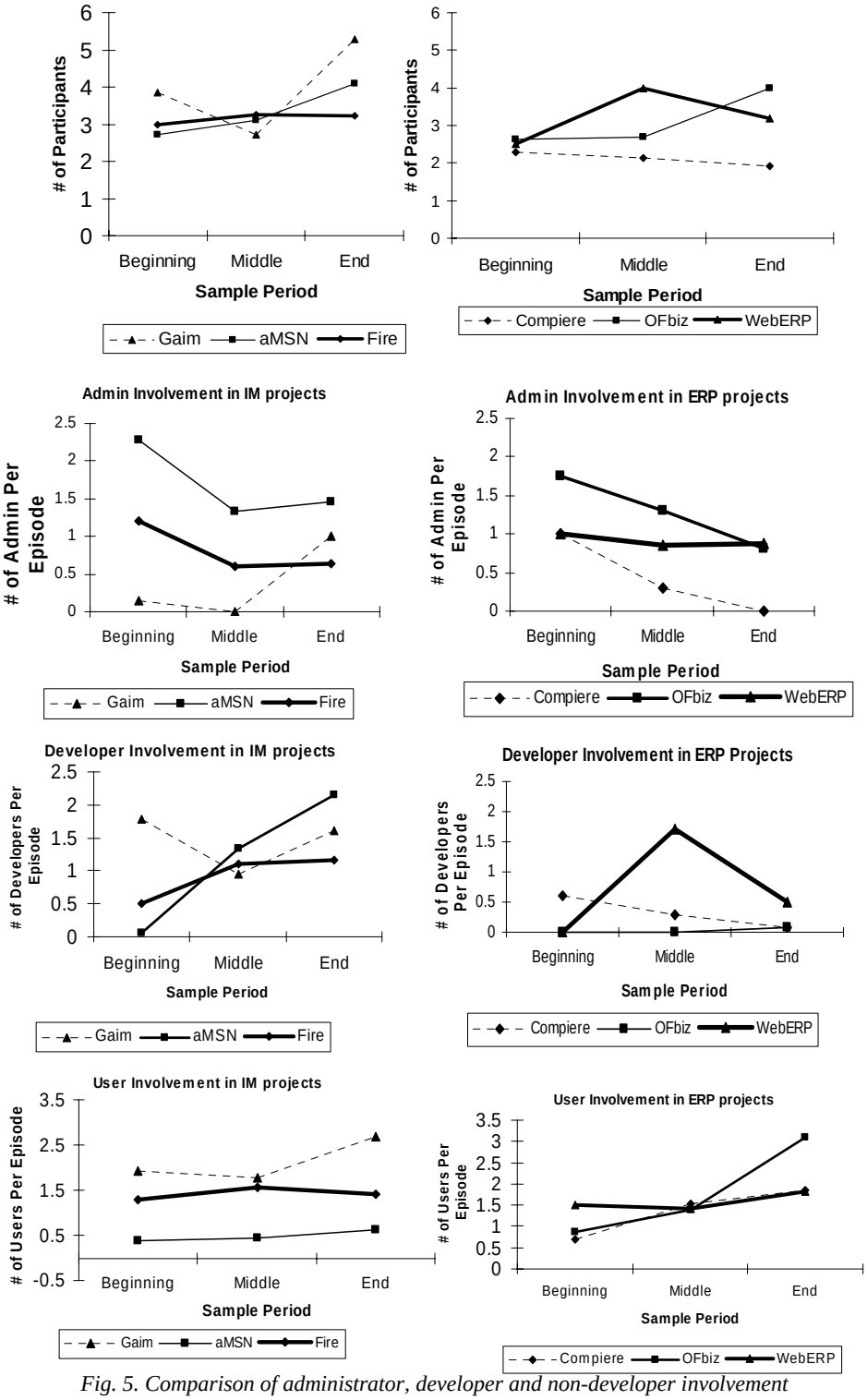

Fig. 5. Comparison of administrator, developer and non-developer involvement

When participation is analyzed in detail, we see that in most projects administrators were highly involved in the initial phase, perhaps to set up the project, communicate to the community and attract project team members (Figure 5). In later periods, the administrators' involvement diminished, allowing the development of a self-governing community. Because of a leadership change, Gaim is an exception to this pattern . In the beginning period, an administrator who was phasing himself out of the project remained relatively silent, allowing developers to actively participate in decision making. Between the middle and end period an active developer became the administrator and remained an active participant in the discussions.

As the administrators became less involved in discussions, the periphery became more involved. However, we see a difference in involvement across project types. In IM projects the developers show a clear pattern of taking more charge in discussions, especially for more effective projects (except for the dip in the middle period for Gaim due to the leadership change). Also, all IM projects show an increase in the non-developer involvement between the first and last periods. The least effective IM project Fire shows an increase in both developer and non-developer involvement for the middle period, but does not sustain this trend towards the end. In the ERP projects, patterns in developer involvement are less clear, however, non-developers are increasingly involved in decisions over time, at an even faster rate than in the IM projects.

These comparisons suggest that although there are similarities across IM and ERP projects, there are some differences in how decision participation evolved over the sampling intervals. Over time, non-developer participation in decision episodes increased for effective projects and administration participation decreased for almost all projects, yet the non-developer and developer participation showed different patterns based on the project type.

4.2 Who Triggers Decision Episodes and Announces Decisions?

In order to better understand the role played by core and peripheral members of the projects in creating decision opportunities for the group, we examined who sent the e-mail message that triggered decision episodes, and who sent the message(s) that announced decisions. Figure 9 shows that both core (administrators plus developers) and periphery played a role in triggering decision opportunities. Figure 6 shows both the IM and the ERP projects from the most effective to the least effective within their software categories. In more effective ERP projects, decision episodes are triggered by the periphery, whereas IM projects show no specific trend. Across all

projects, core members initially create opportunities for decision (i.e., triggers) and in time, this activity moves to the periphery.

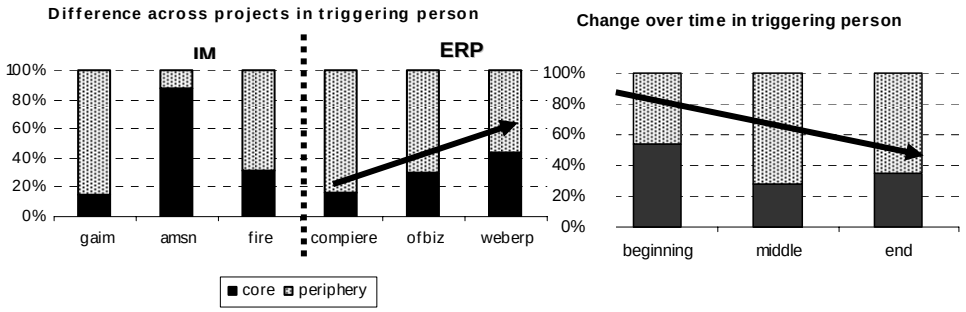

Fig. 6. Comparison of the involvement of the core and periphery in triggering decisions

Figure 7 shows that both core and periphery played a role in announcing decisions. Although across all projects, the members from the core project team announced more decisions, the most effective projects within their categories, i.e., Gaim and Compiere, exhibited higher peripheral involvement in decision announcement than the others. This difference between projects was statistically significant (X^2 =22.038; df=5; p<.01). Across all projects, although most of the decisions were announced by the core, over time, peripheral involvement increased. This difference in peripheral involvement over time was also significant (X^2 =16.204; df=2; p<.01) .

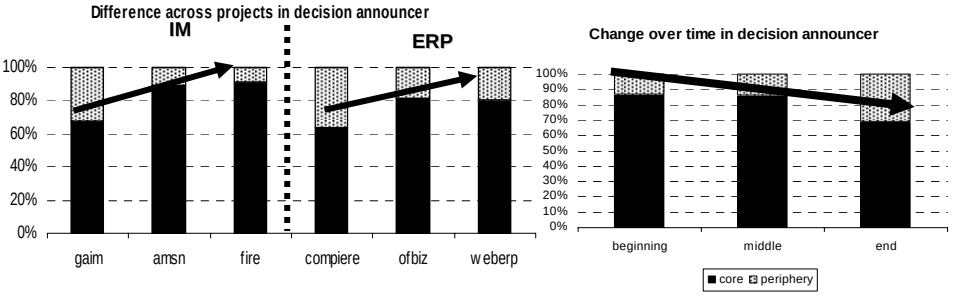

Fig. 7. Comparison of the involvement of the core (administrators and developers) and periphery (users) in announcing decisions

5. DISCUSSION AND CONCLUSION

We presented findings from a long-term research project that seeks to identify work practices that make FLOSS teams effective. This paper described participation in code-related decision-making practices in six open source project teams. In general we found evidence to support the expectation of the literature that greater participation in decision making would be associated with more effective projects.

Generally, among the more effective projects, the number of participants in decision making episodes increased over time, whereas less effective projects either showed a reducing trend or did not sustain initially increasing participation. Similarly, effective projects showed high administrator involvement in decision episodes at the beginning phase of the projects, followed by declining administrator participation coupled with increasing developer and non-developer participation in later periods. (Gaim, although the most effective project in its category, appears to be an exception to this pattern, with low administrator involvement in the first two periods. Gaims pattern is understandable, however, when we recognize that there was a change in administrators during this period. Gaim has very high developer involvement in period 1 may show developers attempting to fill this vacuum.) High administrator involvement in decision making in early phases of these projects may indicate an attempt to establish decision-making standards and norms, while declining involvement may signify increasingly empowered developers and non-developers. Interestingly, less effective projects (Fire, WebERP) show a similar decline in administrator participation in decision making over time, but without a corresponding increase in non-developer involvement.

We also observed higher decision-making participation by the periphery (non-developers) in more effective projects. With one exception (aMSN), a higher percentage of decision opportunities were triggered by non-developers in more effective projects (Figure 9). Similarly, the more effective projects had a higher percentage of non-developers making decision announcements, (Figure 6) even though the core made a higher percentage of decision announcements overall. These two findings–higher overall participation and higher participation by the periphery in more effective projects–supports the notion that increased participation and diversity in decision-making practices is related to improved team performance.

Our findings also suggest that while there are common trends, there are interesting differences between projects in decision-making practices. For example, Compiere showed differences from other projects almost on all measures. This may be because previously it was a closed-source proprietary

company and that the project was not originally self-organizing or emergent. Another such difference can be observed between IM and ERP project categories in the evolution of developer and non-developer participation over time. In IM projects developer participation in decisions grew rapidly, while in ERP projects, this did not occur (Figure 5). Non-developer participation grew more rapidly in ERP projects than in IM projects. This may be explainable by differences in the software type, which in turn affects the type of end-users. IM clients are typically used by individuals, who seek solutions to the issues they face or enhancements to the software for their own needs. On the other hand, the end-users of ERP software are companies, whose own in-house developers play the role of end-users in the FLOSS projects. These end-users are likely to have a high professional interest in the software, and thus high and sustained involvement throughout the project.

By presenting this comparative analysis of the range and evolution of decision-making practices we have begun the process of relating differences in work practices to team effectiveness. These findings suggest that more participative and diverse decision making practices are positively related to team effectiveness in technology supported self organizing teams. The findings also reinforce the idea that all open source teams are not alike. Differences in contextual attributes such as software type and function, as exemplified by the comparison of IM and ERP projects, have an influence on the emergence of work practices. We believe that the variables and relationships we have identified provide the foundation for deeper exploration and potentially richer explanations of the relationships we have described. Future studies should replicate and extend this analysis to additional FLOSS projects, and to other technology supported self organizing teams. A useful future study would be to analyze the decision making process to identify various decision styles by various teams and the relationship between decision styles and team effectiveness.

6. REFERENCES

Bedau, H. (1984). Ethical aspects of group decision making. In W. C. S. a. Associates (Ed.), *Group Decision Making* (pp. 15-150). Beverly Hills: Sage Publications.

Cohen, M. D., March, J. G., & Olsen, J. P. (1972). A garbage can model of organizational choice. *Administrative Science Quarterly, 17*, 1–25.

Dean Jr, J. W., & Sharfman, M. P. (1996). Does decision process matter? A study of strategic decision-making effectiveness (Vol. 39, pp. 368-396): JSTOR.

DeSanctis, G., & Gallupe, B. (1987a). A foundation for the study of group decision support systems. *33*(5), 589–609.

DeSanctis, G., & Gallupe, R. B. (1987b). A Foundation for the Study of Group Decision Support Systems. *Management Science, 33*(5), 589-609.

Eden, C., & Ackermann, F. (2001). Group decision and negotiation in strategy making. *Group Decision and Negotiation, 10*(2), 119-140.

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

Fitzgerald, B. (2006). The transformation of Open Source Software. *MIS Quarterly, 30*(4).

Fjermestad, J., & Hiltz, S. R. (1998/1999). An assessment of group support systems experiment research: Methodology and results. *Journal of Management Information Systems, 15*(3), 7–149.

German, D. M. (2003). The GNOME project: A case study of open source, global software development. *Software Process: Improvement and Practice, 8*(4), 201–215.

Guzzo, R. A., & Salas, E. (1995). *Team Effectiveness and Decision Making in Organizations*. San Francisco: Jossey-Bass.

Hackman, J. R., & Kaplan, R. E. (1974). Interventions into group process: An approach to improving the effectiveness of groups. *Management Science, 5*(3), 459-480.

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

Holloman, C. R., & Hendrick, H. W. (1972). Adequacy of Group Decisions as a Function of the Decision-Making Process. *The Academy of Management Journal, 15*(2), 175-184.

Howison, J., Conclin, M., & Crowston, K. (2006). FLOSSmole: A collaborative repository for FLOSS research data and analysis. *International Journal of Information Technology and Web Engineering, 1*(3), 17-26.

Huber, G. P., & McDaniel, R. R. (1986). The Decision-Making Paradigm of Organizational Design. *Management Science, 32*(5), 572-589.

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

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

Raymond, E. S. (1998b). Homesteading the noosphere. *First Monday, 3*(10).

Turoff, M., Hiltz, S. R., Bahgat, A. N. F., & Rana, A. R. (1993). Distributed group support systems. *MIS Quarterly, 17*(4), 399–417.

Wayner, P. (2000). *Free For All:How Linux and the Free Software Movement Undercut the High-Tech Titans*. New York: HarperCollins.