2001

Shared Mental Models, Familiarity and Coordination: A Mulit-Method Study of Distributed Software Teams

J. Alberto Espinosa
American University

Robert E. Kraut
Carnegie Mellon University

Sandra A. Slaughter
Carnegie Mellon University

Javier F. Lerch
Carnegie Mellon University

James D. Herbsleb
Carnegie Mellon University

See next page for additional authors

Shared Mental Models, Familiarity and Coordination: A Multi-Method Study of Distributed Software Teams

J. Alberto Espinosa, Kogod School of Business, American University
Robert E. Kraut, Human-Computer Interaction Institute, Carnegie Mellon University
Sandra A. Slaughter, Graduate School of Industrial Administration, Carnegie Mellon University
Javier F. Lerch, Graduate School of Industrial Administration, Carnegie Mellon University
James D. Herbsleb, Institute for Software Research International, Carnegie Mellon University
Audris Mockus, Avaya Research Labs

International Conference for Information Systems, ICIS 2002
December 2002
Barcelona, Spain

Shared Mental Models, Familiarity and Coordination: A Multi-Method Study of Distributed Software Teams

Theme: Organization, Culture, Decision-Making & Knowledge

Abstract

Coordination is important in large-scale software development because of the many people involved and the complex dependencies present in software tasks. Even small improvements in productivity can lead to substantial cost-savings and competitive advantage. But despite great technological advances in software engineering and collaboration tools in recent years, coordination in software development projects continues to be problematic. Traditional theories suggest that team members coordinate by programming their tasks and by communicating with each other, but more recent research also suggest that they coordinate through work familiarity and team cognition mechanisms like shared mental models. This paper reports on the results of a multi-method research investigation of how shared mental models, work familiarity and geographic dispersion affect coordination in software teams. This research is based on three studies conducted at a large telecommunications company: face-to-face interviews, survey, and archival studies. Results show that shared mental models have a positive effect on team coordination and that prior familiarity with the same software parts and projects reduces software development time. Results also indicate that geographic dispersion increases software development time and that the effect of work familiarity is stronger for geographically distributed teams than for co-located teams.

Keywords: shared mental models, familiarity, distributed teams, software teams, coordination

Shared Mental Models, Familiarity and Coordination: A Multi-Method Study of Distributed Software Teams

Introduction

Coordination is "the effective management of dependencies among sub-tasks, resources and people" (Malone & Crowston 1994). Coordination is very important in large-scale software development because large software products and their development processes contain complex dependencies (Crowston & Kammerer 1998, Curtis et al. 1988, Kraut & Streeter 1995) and involve many teams and individuals working in parallel on the same product. But despite today's advanced software engineering processes and tools, coordination continues to be problematic in this domain, largely because of human and behavioral factors (Bohem 1981, Brooks 1995, Curtis et al. 1988). Also, because of the need to find large numbers of skilled and specialized developers, and because of the ease with which software parts can be exchanged through networks, large-scale software development often involves people collaborating from more than one geographic location. While this has some benefits, geographic dispersion brings increased coordination overhead and more substantial delays in software development (Herbsleb et al. 2001; Herbsleb et al. 2000; Carmel 1999). Consequently, it is very important that we understand which factors contribute to coordination in this domain, and how distance affects a software team's ability to coordinate.

Large-scale software development projects are often supported by sophisticated workflow and collaboration tools like configuration management systems. However, some studies have found that software engineering tools only go so far in helping these teams increase their productivity, and that when team members have familiarity with their application domain and shared knowledge of the task and each other they are more coordinated and perform better (Curtis et al. 1988, Walz et al.

1993; Faraj & Sproull 2000; Crowston et al. 1998). This "organized knowledge that members share about things like the task, each other, goals and strategies" is also referred to as "shared mental models" (Cannon-Bowers et al. 1993, Klimoski & Mohammed 1994). It has been argued in the team cognition literature that these models help team members develop accurate explanations and expectations about the task and members' behavior, which in turn helps them coordinate implicitly. A related is "work familiarity", which is defined as "the specific knowledge that members have about aspects of the workplace (e.g., machinery, materials, task environment, people)" (Goodman & Shah 1992). Studies have found that familiar workers have higher levels of performance (Goodman & Shah 1992; Goodman & Leyden 1991; Goodman & Garber 1988). It has been argued in this literature that work familiarity is more beneficial in tasks with high levels of uncertainty and complexity such as large-scale software development.

While many have suggested that shared mental models are beneficial for team performance, most of the related research has focused on real-time, co-located teams working on synchronous tasks, with very little empirical evidence in other contexts (e.g., software tasks), particularly when teammates are separated by time (i.e., asynchronous) or distance. This research contributes to fill this gap. Therefore, our ***research question*** is:

How do shared mental models, work familiarity and distance affect coordination in large-scale software development?

Shared Mental Models

The classic organizational research literature suggests that team members coordinate via task programming mechanisms (e.g., plans, specifications) or by communicating (March & Simon 1958, Thompson 1967, VanDeVen et al. 1976). But the team cognition research literature suggests that mechanisms like shared mental models also aid coordination. A recent study of teams working

in a flight simulation task found that shared mental models had a positive effect on team processes (e.g., coordination), which improved performance (Mathieu et al. 2000). Studies of software teams have also found that their team members need to acquire, share and integrate substantial amounts of knowledge of the application domain to ensure positive outcomes in software projects (Curtis et al. 1988, Walz et al. 1993). Similarly, a recent study of software developers found that knowing where expertise resided in their teams had a positive effect on performance (Faraj & Sproull 2000). Yet another study with software requirement analysis teams found that teams that exhibited a "collective mind"—i.e., a shared understanding of the group's task and each other (Weick et al. 1993)—were more coordinated because members understood how their work contributed to group outcomes (Crowston et al. 1998).

Work Familiarity

Shared mental models and work familiarity are related, but conceptually distinct constructs. The shared mental models are based on knowledge similarity within a team while work familiarity refers to knowledge that individuals members posses. Shared mental models among team members develop from working and training together on similar tasks (Levesque et al. 2001, Mathieu et al. 2000, Rentsch et al. 1994) and from things like team experience and common organizational familiarity (Rentsch et al. 2001). Consequently, team members with work familiarity in similar tasks are more likely to have stronger shared mental models. We argue in this paper that team members that are familiar with the same parts of the software code and with the same software projects have more organized shared knowledge (i.e., stronger shared mental models) than those team members that are familiar with different parts of the software code and different software projects.

While there is evidence suggesting that familiar workers are more productive (Goodman & Shah 1992), we argue that in order for work familiarity to help coordination (i.e., manage dependencies) team members need to have work familiarity in similar aspects of the task. For example, a study with airline pilots found that crews that had recently flown together performed better and that familiarity between crewmembers made their communication and information exchange more effective, and helped them tailor behaviors and interaction to particular situations, which reduced error rates (Kanki & Foushee 1989). Two software developers who are familiar with different application domains and different software projects may possess individual knowledge to develop software competently. However, two software developers who have work familiarity with the same software processes, application domains and projects, will not only be able to perform well individually, but will also have more common ground in their communications and more organized shared knowledge, which will in turn help them communicate more effectively and anticipate future states of the task, thus helping them coordinate. Thus, we posit that,

Hypothesis 1: *Shared mental models and similar work familiarity have a positive effect on coordination in large-scale software development*

Distance

There is some empirical evidence suggesting that distance makes it more difficult to coordinate software tasks (Curtis et al. 1988, Herbsleb & Grinter 1999, Carmel 1999) and that developing software across locations takes longer (Herbsleb et al. 2001). Distance makes it more difficult for team members to coordinate their work, even if they are only a few feet away from each other (Allen 1977) because distributed teammates don't have the benefits of spontaneous and frequent interaction and their communication media is not very rich. Therefore, it takes longer to obtain responses and correct miscommunication. This suggests that,

Hypothesis 2: *Co-located teams are more coordinated than distributed teams*

Distance changes the task context and impacts the effectiveness of different coordination mechanisms. Team interaction in distributed contexts tends to be less frequent and less spontaneous, affecting members' ability to coordinate through communication (Curtis et al. 1988, Kraut & Streeter 1995). Also, these teams often don't have the benefit of a shared workspace or familiarity with the local work environment in other sites, which reduces common grounding in their communication. Therefore, shared mental models and similar work familiarity can help distributed members compensate for their less effective communication and information exchange. On the other hand, co-located team members can communicate effectively when they need to coordinate, so the incremental benefits of shared mental models and similar work familiarity may not be as great as for distributed teams. Therefore,

Hypothesis 3: *The effects of shared mental models and similar work familiarity on coordination are stronger for geographically distributed teams than on co-located teams*

Research Methodology

To evaluate our hypotheses, we conducted a field study in two separate divisions at a large international telecommunications company. One division produces software for the European GSM (Global System for Mobile communications) telephony industry and the other division produces software for telephony switching equipment. Both divisions are quite large and global employing several thousand software developers and support staff. We adopted a multi-method approach involving three separate but related studies, thus providing methodological and data triangulation. We used a "two-phase sequential" (i.e., qualitative-quantitative) approach (Tashakkori & Teddlie

1998) in which results from the first phase informed the design of the second phase. The first phase involved an interview study undertaken to: gain familiarity with the company's software context and its coordination challenges; evaluate and refine the theoretical framework; identify an effective unit of analysis; and define the main variables for the study. The second phase involved two parallel studies, one of which was based on data collected from a survey of software developers from a telecommunications company, and it examined the effects of shared mental models on coordination, controlling for the use of other coordination mechanisms. The other study examined archival data from related software production records, and was conducted to investigate the effects of similar work familiarity (i.e., a proxy measure for shared mental models) on software development productivity to help validate findings across studies.

Interview Study

This study was conducted in two European countries through face-to-face, semi-structured interviews of 36 developers and managers from a software "release team" for wireless telephony equipment, which was responsible for the implementation of a number of new features (i.e., a release or version) to the switch's software. The team selected for the study had approximately 50 software professionals, which had completed over 70% of the release's software, so its members had memories of recent relevant experiences to discuss during the interviews. Most of the release's software was developed at the two locations selected. The interview transcripts were first analyzed to uncover general recurrent themes, which were then used to produce a codebook for thematic coding (King 1998). The hierarchical coding scheme adopted had three main code categories and several related sub-codes. The first code category identified attributions about the effect of different team cognition mechanisms on coordination. Attributions were coded when a participant indicated that a particular mechanism was important (or that its absence was detrimental) for coordination.

The second code identified specific issues discussed for each context: co-located (e.g., staff overload) and cross-site work (e.g., little opportunity for interaction). The final code categorized instances of coordination problems into more general types of coordination problems. The interview transcripts were then coded by the researcher and by an independent rater (Cohen's Kappa = 72.1%).

Results

Many participants (78%) attributed their coordination to shared mental models of the team (e.g., knowing who knows what, familiarity with colleagues) and many (75%) attributed it to shared mental models of the task (e.g., shared knowledge of concepts and products; common vision of goals), supporting Hypothesis 1. However, results also revealed differences between co-located and geographically distributed work. More participants (78%) attributed shared mental models of the team as important for geographically distributed work than for co-located work (13%), and many (69%) indicated that having prior knowledge of colleagues in other sites helped offset problems associated with distance. In contrast, more participants (63%) attributed shared mental models of the task as important for co-located work than for geographically distributed work (31%), providing mixed support for Hypothesis 3. The importance of shared task knowledge appears to be more salient in co-located work while the importance of knowing colleagues' expertise and skills appears to be more salient in multi-site work. A possible explanation is that while shared task knowledge may be important for distributed work, if team members don't know their colleagues at other sites they can't know how much task knowledge they have in common.

Results also show that coordination is less problematic for co-located than distributed developers. Many participants (53%) explicitly mentioned that they did not see many coordination problems with co-located work because they knew their local colleagues well and because they

interacted frequently with them. In contrast, the majority (91%) mentioned problems affecting geographically distributed work (e.g., little familiarity with colleagues, no presence awareness, few opportunities for interaction), supporting Hypothesis 2.

Survey Study

The survey study explored the effect of shared mental models on coordination using data collected with a web survey. Participants responded to items about specific software modifications in which they were involved, and about each developer that collaborated in those modifications. The survey instrument included items on the use of task programming mechanisms, team communication, shared mental models and coordination. Most shared mental models (SMMs) can be classified as knowledge similarity about the task or about the team (Cannon-Bowers et al. 1993, Klimoski et al. 1994, Rentsch & Hall 1994). Thus, we measured SMM of the task (i.e., knowledge that team members share about the task) and SMM of the team (i.e., knowledge that team members share about each other).

The participants in this study developed software for digital telephony switching equipment. This equipment runs complex real-time software containing several million lines of code. The software is updated with incremental releases (i.e., versions) through "modification requests" (MRs). A "Change Control Board" reviews all MRs and assigns priorities to the ones it approves. All software developed is traceable to a formally approved MR, which is implemented by relatively small MR teams (2-12 developers), making these teams effective units of analysis for this study. An MR contains one or many "Deltas"—i.e., the addition, deletion and/or modification of one or many lines of code on a single file by a single developer.

The survey was administered in two sites (U.S. and England) where over 90% of the code for a main sub-system of this switching equipment had been developed. The web survey was

dynamically generated for each respondent with questions about one to three MRs in which the respondent recently contributed Deltas, and about each developer who collaborated in these MRs. A total of 113 participants were identified and 97 of them completed the survey (85.8% response rate), providing usable data for 54 MRs. MR teams in the sample ranged in size from 2 to 7 with an average of 3.5 members. Variables computed at the individual level were averaged to the team level—analysis of intra-class correlation and within-team agreement (James et al. 1984) statistics justified the aggregation of individual-level variables to the team level. The main measures used in this study are described in Appendix A.

Results

We used weighted least squares (WLS) regression with a general weighting procedure to correct for non-spherical residuals. The beta values indicated below are standardized regression coefficients. Results provided partial support for Hypothesis 1. SMMTeam had a positive effect on team coordination (β=0.089, p=0.035), but SMMTask was not significant. We attribute this to the positive effect of the use of the software configuration management system (SCMS) on coordination (β=0.295, p=0.008). Developers use this system to manage the integration of multiple versions of the software and to record MR technical details and problems (e.g., error logs, developer comments). It appears that the SCMS helped these technical teams coordinate their work through communication, so shared task knowledge did not provide incremental benefits for these small technical teams. Surprisingly, neither the difference in coordination between geographically distributed teams and co-located teams, nor the interactions between distance and each of the shared mental model variables were significant. These results fail to support hypotheses 2 and 3 about the effects of geographic dispersion.

Archival Study

This study explored the effect of similar work familiarity on software development time using data from the SCMS database, which records data for each Delta and MR implemented. Shared mental model variables cannot be constructed from archival software production data. However, similar work familiarity between any two members in a team can be computed by counting things like the number of files and modules in which both team members have developed software, and the number of software projects in which both team members have collaborated. A similar work familiarity variable can be computed for a software team by averaging the similar work familiarity of each dyad in the team. A team whose members have more similarity in their work familiarity is more likely to have stronger shared mental models than a team in which members have individual work familiarities in dissimilar aspects of the task. For consistency with the survey study, the unit of analysis in the archival study was also MR coding teams for the same software product. The sample came from all MRs implemented in the prior three years in which two or more developers were involved, yielding data for 54,665 Deltas for 1,170 MRs. The main measures used in this study are described in Appendix B.

Results

Development time intervals were substantially longer for geographically distributed MRs (97.3 days) than for co-located MRs (48.3 days) ($p=0.003$), supporting Hypothesis 2. Further analysis was conducted using Weighted Least Squares (WLS) regression models to correct for non-spherical residuals. The beta values indicated below are standardized regression coefficients. Three models were estimated in a hierarchical fashion to study incremental effects. In the baseline model, which does not include the familiarity variables, we found that team experience significantly reduced software development time ($\beta= -0.288$, $p<0.001$). The work familiarity variables were

added to the model next. Familiarity with the same modules/files ($\beta = -0.067$, $p<0.001$) and on the same MRs ($\beta = -0.046$, $p<0.001$) reduced software development time, supporting Hypotheses 1. Interestingly, the team experience coefficient became non-significant ($\beta = -0.090$, $p=0.137$) when the familiarity variables were added. This result suggests that while developer experience is an important factor in reducing software development time, joint developer experience with the same MRs, modules and files is more important when multiple developers are involved in a MR implementation. We then added to the model interaction variables of geographic dispersion with each of the two familiarity variables. The significance levels of both interaction variables of distance with familiarity with the same modules/files ($\beta = -0.027$, $p=0.034$) and on the same MRs ($\beta = -0.023$, $p=0.073$) were moderate. Their negative signs support Hypotheses 3 that similar work familiarity helps reduce software development time, but more so when the work is done across sites.

Discussion

Results are consistent across the three studies. They support the main hypothesis of this research that shared mental models help coordination in large-scale software development. Furthermore, coordination and software development time were negatively correlated ($r = -0.39$, $p=0.011$), providing some assurance of convergent validity for the coordination measure used in the survey study, and establishing a link between coordination success and software development time (i.e., survey and archival studies). Familiarity with the same modules/files was also positively correlated with SMM of the task ($r=0.304$, $p=0.036$) and SMM of the team ($r=0.285$, $p=0.052$). While the significance levels of these results are only moderate (partly due to the sample size of the survey study), these results provide some validity for the measures used for shared mental models in the survey study.

Hypothesis 2 that distance has a negative effect on coordination was also well supported in the interview and archival studies, but not in the survey study. We attribute this lack of support in the survey study to the small number of observations available for distributed work, which may have been insufficient to reveal its effects with sufficient statistical power. This also suggests software work in this organization is more likely to be distributed across sites under special circumstances (e.g., need to service a remote client). In contrast, most of the hypothesized effects of distance were clearly supported in the interview and archival studies. Consistent with prior studies, it is more difficult to coordinate and it takes longer to develop software when working from multiple locations. Hypotheses 3 that the effects of shared mental models are stronger for geographically distributed teams than for co-located teams were supported in the interview and archival studies, but not in the survey study. Again, we attribute this to the same reasons articulated earlier. However, the negative interaction effects of distance and familiarity support our argument that when software teams work across sites they do not have the benefit of frequent, rich and spontaneous communication, and therefore benefit more from shared knowledge developed from their similar work familiarity.

Consistent with the software engineering literature (Bohem 1981, Brooks 1995), we found that individual experience in MR teams helps reduce software development time. But, interestingly, and consistent with recent empirical evidence suggesting that sharing and integrating individual knowledge is important (Crowston et al. 1998, Curtis et al. 1988, Faraj et al. 2000, Walz et al. 1993), we found that when multiple software developers collaborated in MRs, having team members with similar prior experience was more effective in reducing software development time, particularly when the software was developed from multiple sites.

Conclusions

This study has a number of potential limitations. Some of the findings may be limited in scope to the company we studied and the methods we used. The survey study relied on self-reports, although we reduced potential problems of common method variance by measuring the dependent variable from responses about MRs, while constructing the main independent variables (i.e., shared mental models, team communication) from responses at the dyad level. The survey study was also limited because of the small sample size, which was imposed by the logistical difficulties of collecting team data from the different software groups involved. Finally, our focus in the survey and archival studies is on MR coding teams. Further research is needed to study teams working on other development phases (e.g., designers, testers).

Nevertheless, this study makes a number of important contributions. Much of the prior research on shared mental models has been theoretical and most of the related empirical work has taken place in laboratory experiments using real-time (i.e., synchronous) simulated tasks. Similarly, most of the empirical research on familiarity has been conducted using synchronous tasks. This study contributes to the empirical body of literature on shared mental models, familiarity, and coordination in asynchronous and distributed collaboration. It is also the first to link the concepts of shared mental models and familiarity, and the first to combine traditional organizational theories of coordination with team cognition theories into a single theoretical framework of team coordination that includes geographic distance. Finally, ours is one of the few empirical studies that investigated the effect of shared mental models and familiarity on coordination in a specific, asynchronous, real organizational context in the software domain, using multiple research methods and data sources.

References

Allen, T. (1977). *Managing the Flow of Technology*. Cambridge, MA: MIT Press.

Bohem, B. R. (1981). *Software Engineering Economics*. Englewood Cliffs, N.J.: Prentice-Hall, Inc.

Brooks, F. (1995). *The Mythical Man-Month: Essays on Software Engineering* (Anniversary ed.): A. Wesley.

Cannon-Bowers, J. E., Salas, E., & Converse, S. (1993). "Shared Mental Models in Expert Team Decision-Making". In Castellan, J. (Ed.), *Individual and Group Decision-Making: Current Issues*: 221: LEA Publishers.

Carmel, E. (1999). *Global Software Teams*. Upper Saddle River, NJ: Prentice Hall.

Cooke, N. J., Salas, E., Cannon-Bowers, J. A., & Stout, R. J. (2000). "Measuring Team Knowledge". *Human Factors*, 42(1): 151-173.

Crowston, K. & Kammerer, E. E. (1998). "Coordination and Collective Mind in Software Requirements Development". *IBM Systems Journal*, 37(2): 227-245.

Curtis, B., Krasner, H., & Iscoe, N. (1988). "A Field Study of the Software Design Process for Large Systems". *Communications of the ACM*, 31(11): 1268-1286.

Faraj, S. & Sproull, L. (2000). "Coordinating Expertise in Software Development Teams". *Management Science*, 46(12): 1554-1568.

Goodman, P. S. & Shah, S. (1992). "Familiarity and Work Group Outcomes". In Worchel, S. & Wood, W. & Simpson, J. A. (Eds.), *Group Processes and Productivity*: 578-586. Newbury Park, CA: Sage Publications.

Goodman, P. S. & Shah, S. (1992). "Familiarity and Work Group Outcomes". In Worchel, S. & Wood, W. & Simpson, J. A. (Eds.), *Group Processes and Productivity*: 578-586. Newbury Park, CA: Sage Publications.

Goodman, P. S. & Leyden, D. P. (1991). "Familiarity and Group Productivity". *Journal of Applied Psychology*, 76(4): 578-586.

Goodman, P. S. & Garber, S. (1988). "Absenteeism and Accidents in a Dangerous Environment: Empirical Analysis of Underground Coal Mines". *Journal of Applied Psychology*, 73(1): 81-86.

Herbsleb, J., Mockus, A., Finholt, T., & Grinter, R. E. (2001). "An Empirical Study of Global Software Development: Distance and Speed". Paper presented at the 23rd. International Conference on Software Engineering (ICSE), Toronto, Canada.

Herbsleb, J. D., Mockus, A., Finholt, T., & Grinter, R. E. (2000). "Distance, Dependencies and Delay in a Global Collaboration". Paper presented at the CSCW 2000, Philadelphia, Pennsylvania.

Herbsleb, J. D. & Grinter, R. E. (1999). "Architectures, Coordination, and Distance: Conway's Law and Beyond". *IEEE Software*(September/October): 63-70.

James, L. R., Demaree, R. G., & Wolf, G. (1984). "Estimating Within-Group Interrater Reliability With and Without Response Bias". *Journal of Applied Psychology*, 69(1): 85-98.

Kanki, B. G. & Foushee, H. C. (1989). "Communication as Group Process Mediator of Aircrew Performance". *Aviation, Space, and Environmental Medicine*.

King, N. (1998). "Template Analysis". In Symon, G. & Cassell, C. (Eds.), *Qualitative Methods and Analysis in Organizational Research*: 118-134. Thousand Oaks: Sage Publications.

Klimoski, R. J. & Mohammed, S. (1994). "Team Mental Model: Construct or Methaphor". *Journal of Management*, 20.

Kraut, R. E. & Streeter, L. A. (1995). "Coordination in Software Development". *Communications of the ACM*, 38(3): 69-81.

Levesque, L. L., Wilson, J. M., & Wholey, D. R. (2001). "Cognitive Divergence and Shared Mental Models in Software Development Project Teams". *Journal of Organizational Behavior*, 22(March): 135-144.

Malone, T. & Crowston, K. (1994). "The Interdisciplinary Study of Coordination". *ACM Computing Surveys*(March).

March, J. & Simon, H. (1958). *Organizations*: John Wiley and Sons.

Mathieu, J., Goodwin, G. F., Heffner, T. S., Salas, E., & Cannon-Bowers, J. A. (2000). "The Influence of Shared Mental Models on Team Process and Performance". *Journal of Applied Psychology*, 85(2): 273-283.

Rentsch, J. R. & Hall, R. J. (1994). "Members of Great Teams Think Alike: A Model of the Effectiveness and Schema Similarity Among Team Memebers". *Advances in Interdisciplinary Studies of Work Teams*, 1: 223-261.

Rentsch, J. R. & Klimoski, R. J. (2001). "Why do Great Minds Think Alike?: Antecedents of Team Member Schema Agreement". *Journal of Organizational Behavior*, 22(March): 107-120.

Tashakkori, A. & Teddlie, C. (1998). *Mixed Methodology: Combining Qualitative and Quantitative Approaches*. Thousand Oaks, California: Sage Publications.

Thompson, J. (1967). *Organizations in Action*: McGraw-Hill.

VanDeVen, A. H., Delbecq, L. A., & Koenig, R. J. (1976). "Determinants of Coordination Modes Within Organizations". *American Sociological Review*, 41(April): 322-338.

Walz, D. B., Elam, J. J., & Curtis, B. (1993). "Inside a Software Design Team: Knowledge Acquisition, Sharing, and Integration". *Communications of the ACM*, 36(10): 63-77.

Weick, K. (1993). "The Collapse of Sensemaking in Organizations: The Mann Gulch Disaster". *Administrative Science Quarterly*, 38(4): 628-652.

Appendix A

Measures Used in Survey Study

Measure	Description	Notes
Coordination	Ten survey items based on coordination problems uncovered in the interview study (e.g, "we had a lot of integration problems with this MR", "we often had to do re-planning because of missed delivery dates and delays associated with this MR").	Cronbach-α=0.919
Shared Mental Model of the Task (SMMTask)	Consistent with prior studies, we asked participants to rate task-relevant items about specific MRs in which they worked (e.g., "I was never certain whether the code written for this MR would need further re-work"), and then computed the within-team agreement score (R_{WG}).	Cooke et al. 2000, Levesque et al. 2001, Mathieu et al. 2000, Rentsch et al. 2001, James et al. 1984
Shared Mental Model of the Team (SMMTeam)	Similar to SMMTask, but we asked participants to rate statements about each team member who worked on these MRs (e.g., "This teammate's knowledge of specifics about the MR was very high").	Same as SMMTask
Geographic Dispersion (0,1)	1 if the team operated in more than one location; 0 if the team operated from a single location	Most distributed teams operated in only two locations
Task Programming	Use of a software configuration management system and other software tools	
Team Communication	Communication frequency with each teammate in the MR through different media (e.g., e-mail, face-to-face) and aggregated to the team level.	Cronbach-α=0.844
Other control variables	Member dependency (2 items), team size and MR priority	

Appendix B

Measures Used in Archival Study

Measure	Description	Notes
Software Development (Elapsed) Time	A proxy measure for team coordination—i.e., better coordinated software projects are more likely to be implemented in shorter time intervals, other things being equal. Calculated as the time difference between time stamps for the first and last deltas of the MR. The variable was transformed by taking logarithms to approximate a normal distribution.	
Similar Work Familiarity	Similar work familiarity metrics were first computed for ***each dyad*** in a MR team: familiarity with the same modules/files (number of modules/files in which both dyad members had developed software) and familiarity with the same MRs (number of prior MRs in which both dyad had collaborated). Familiarity measures with modules, files and MRs were reduced to two factors using factor analysis: one factor loaded highly on the familiarity metrics for modules and files (0.947 and 0.811 respectively), explaining 72% of the variance; the second factor loaded highly on the familiarity metric for MRs (0.961), explaining an additional 20% of the variance, for a total of 92% of the variance explained. The dyad measures were then averaged across all dyads in the MR team. Intra-class correlation and within-team agreement statistics indicated that it was appropriate to aggregate dyad data to the team level.	James et al. 1984
Geographic Dispersion (0,1)	1 if the team operated in more than one location; 0 if the team operated from a single location	Most distributed teams operated in only two locations
Other control variables	Variables that may affect software development time: MR size, MR complexity, MR team size, MR priority and team's domain experience.	