Differential and Depreciation Effects of Shared Experience: Evidence from Software Projects

Keumseok Kang
Florida International University
kskang@fiu.edu

Inkyoung Hur
Florida International University
ihur001@fiu.edu

Abstract

Shared experience (i.e., experience working together) among project team members is known to affect a project team's performance. Yet, little is known of whether the effects of shared experience homogeneous. We empirically find that the effects of shared experience on a project's performance vary by the outcome of shared experience (e.g., successful vs. unsuccessful) as well as project team role (e.g., project managers vs. non-managers) in the context of software development. Moreover, we find the effects of shared experience depreciate over time. Our study has a theoretical contribution by showing differential and depreciation effects of shared experience. It also has practical implications on the project staffing problem.

1. Introduction

Software development is an interdependent group task where a group of individuals with different knowledge and skills work together. Large-scaled software applications like an enterprise information system are usually developed by a group of software engineers which is called a software project team. Due to the interdependent nature of this group task, interrelationship and teamwork among project team members have been considered as one of the most influencing factors determining the success of a software project in the literature [e.g., 9, 20, 22, 23]. As knowledge and skills possessed by software engineers become more specialized and remote and virtual software development environments are more prevalently used, interrelationship and teamwork among project team members get more important, and how to build and sustain a good interrelationship and teamwork within a project team becomes one of the main managerial issues of software organizations.

Prior studies evidenced that shared experience among project team members (i.e., sometimes called experience working together or team familiarity) positively affects on building and sustaining a good interrelationship and teamwork within a project team and eventually increases project performance [e.g., 9, 12, 15, 20, 21, 22, 23]. Although these prior studies have important implications, there are still important yet unexplored areas that deserve further research so as to provide more substantive guidance for software project management. In this paper, we extend this stream of research by investigating three important aspects of shared experience and evidence them in the context of software projects.

First, prior studies overlooked the outcomes of projects where focal team members have worked together before. Not all projects are necessarily successful. The question is whether or not the effects of shared experience in a successful project and in an unsuccessful project are same. A project team may still build a transactive memory (i.e., knowledge about who knows what within a project team), which creates the positive effects of shared experience on performance [18], even in failed projects. However, a failure may deteriorate trust among team members, while trust is known as another mechanism to show how shared experience increases teamwork and performance [15].

Second, previous research did not consider the project team roles and their relationship in a project team. In general, a project team is not a flat organization, but has a hierarchy. A typical project team consists of one or multiple project managers and their subordinate non-managers (e.g., designer, developer, and tester in the context of software development). The question is whether or not the effects of shared experience between different project team roles (e.g.., between project manager and non-manager, between project managers, and between non- managers) are same. Faraj and Sproull [9] and Reagans et al. [15] and Kanawattanachai and Yoo [21] argued that shared experience develops a coordination ability based on a transactive memory – assigning a task to a team member who knows well about the task and resolving potential conflicts between team members. This coordination ability does not seem to be equally necessary to all team members. Due to the different nature among project

1530-1605/12 $26.00 © 2012 IEEE
DOI 10.1109/HICSS.2013.178

IEEE computer society

team roles, this coordination ability is more required to project managers than non-managers.

Third, it is known that cumulated knowledge (gained from experience) does not persist forever in general [7]. Therefore, cumulated shared experience itself may depreciate over time (e.g., forgetting about who knows what). Also, cumulated knowledge of individual members may change (i.e., individual experience may depreciate or be cumulated) over time. This makes shared experience obsolete, too.

The remainder of this paper is organized as follows. In the next section, we present our theoretical background and develop research hypotheses. Next, we describe the research context and outline the analysis approach. Then we report our preliminary analysis results and discuss the results with concluding remarks.

2. Theory and Hypotheses

If project team members have shared experience (i.e., prior experience working with each another), since they might have already established multiple communication channels and perceived the personal characteristic or professional abilities possessed by each other, they should reduce mishaps in communication and collaboration, predict potential risks better, and eventually should do their jobs better. There are previous studies supporting this argument. Gruenfeld et al. [11] found that a project team consisting of individuals who have prior experience of working together with each other solves a complex problem better than a team consisting of those who have not worked together before. Faraj and Sproull [9] found that it is not an amount of expertise cumulated within a team but an ability to coordinate and distribute expertise across team members which leads a success of software project. Faraj and Sproull [9] defined the coordinating ability to locate an expertise within a team, to know where the expertise is needed, and bring the needed expertise appropriately. Reagans et al. [15] explained the mechanism for the positive effects of shared experience on performance using coordinating ability, similar to Faraj and Sproull [9], as well as increased trust among team members. Boh et al. [4] found that shared experience positively affects performance at multiple levels (i.e., individual, group, and organization) and further showed that the effects of shared experience on performance at the individual level is different from those at the group and organization levels, where activities are more interrelated compared to the individual level. Based on the results of the previous literature, it is naturally anticipated that sharing experience among project team members affects the performance of the focal project team. Thus we hypothesize:

H1: Shared experience among project team members positively affects the performance of the project team.

However, previous literature lacks of explanation about the impacts of the outcome of prior experience working together on current group work. In reality, a number of software projects actually fail for several reasons [16]. When it comes to the ability to coordinate expertise, it seems like that the outcomes of shared experience (i.e., a success or failure of prior project where team members worked together before) do not matter significantly because even in failed projects, a project team may still develop the coordinating ability. For example, even if a project fails because of insufficient ability to coordinate expertise, project team may still learn about who has which expertise and where and when to use this expertise because every knowledge, even including expertise coordination ability, can be acquired by trials and errors. However, there is another factor which enables shared experience to positively affect performance, besides the expertise coordination. It is trust among project team members. Reagans et al. [15] argued that shared experience increases the level of trust among working partners and in turn this cumulated trust facilitates sharing knowledge and skills among them. If a project fails due to the conflicts between team members among other things, then this experience may hurt trust among members who have worked together, while it may still increase the expertise coordination ability. Another possible scenario could be a failure may be due to lack of expertise – i.e., there is really no sufficient expertise required to perform tasks within a team. If it is the case, any coordinating ability may not be gained from experience working together because the team would have no chance to experience how to locate and coordinate expertise. For these reasons, we believe the outcomes of prior shared experience among project team members should affect the effects of that shared experience on performance. Thus we hypothesize:

H2: The effects of shared experience in successful projects are greater than those in unsuccessful projects.

Reagans et al. [15] and Faraj and Sproull [9] emphasized the importance of the ability to

coordinate expertise in a group work such as software development and medical surgery. The expertise coordinating ability is defined as 'knowing who knows what' [9], which is alternatively called transactive memory in the literature [18]. This ability is necessary to every member in a project team [9]; however, considering the fact that various specialized project team roles are practiced in most contemporary project teams, this ability may not be equally required to all project members. Without loss of generality, we classify project team roles into two types – project managers and those who are not project managers which we call non-managers. The coordinating ability based in a transactive memory (i.e., knowing who knows what within a project team) seems to be more essential to project managers than non-managers because it is project managers who should know and coordinate expertise possessed by team members across the project team. Assigning tasks to the most competent persons with that tasks and coordinating interrelated tasks among team members is not a main role of non-project managers but certainly the responsibility of project managers. Thus we hypothesize:

H3: The effects of shared experience between project managers and non-managers are greater than those between non-managers and those between project managers.

Although repeated experience accumulates knowledge, knowledge may depreciate over time. Since Ebbinghaus' first documentation of knowledge depreciation in 1885 [7], knowledge depreciation effects (sometimes called forgetting effects) have been examined at the individual level [2, 10, 14], as well as at the organization level in such various contexts as aircraft production [3], automotive assembly [8], ship building [1,17], textile manufacturing [14], and pizza franchises [6]. What causes knowledge depreciation? A number of researchers found that knowledge depreciation occurs when there is an interruption or a prolonged break in the learning process [2, 3, 7]. Some studies have also suggested that knowledge may naturally depreciate over time even in the absence of a break [17]. Tremendous evidences of knowledge depreciation in the literature naturally lead us to anticipate that depreciation should occur with knowledge cumulated from shared experience (i.e., experience working together). Project team members may forget who know what within their project team as they forget task knowledge cumulated from their individual experience. Moreover, as mentioned earlier, expertise of individual members may change. For example, individual experience may be depreciated (i.e., forgotten) or be cumulated (i.e., learned from individual experience or from individual training) over time. Such changes quickly make shared experience obsolete, too. Thus we hypothesize:

H4: The effects of shared experience depreciate over time.

3. Study Method

3.1. Data

We have collected and analyzed an extensive archival data set from a prominent international IT service company in Korea. The company provides contract-based custom software development and software maintenance for a variety of applications to a broad range of industries. The company has practiced software development according to global standards – it has been certified by ISO 9001 and the Capability Maturity Model (CMM) since the 1990's and has acquired Capability Maturity Model Integration (CMMI) Level 5 in 2004. It has maintained detail project data since the late 1980s.

Our data set contains detailed information on software development projects (e.g., project schedule, plan, performance, customer, industry, application type, etc.), the employees who worked on those projects, and the technologies and methodologies used in those projects. Our sample includes 497 software development projects that ended between 2005 and 2007. However, historical archives of all projects dating back to the company's founding in the late 1980s were used to capture the employees' prior shared experiences.[1] Most projects were large-scale development projects. The average project team size was approximately 10 developers, and the average project duration was 10 months.

3.2. Dependent Variable: Project Effort

We measure software development performance of a software development project using the actual labor effort (*Effort*) in hours. This measure has been widely used in prior literature [4].[2] The effects of shared

[1] Due to the limited availability of some variables for older projects, we could not use all historical projects for the data points for our regression.

[2] While other dimensions of software development performance (e.g., on-time delivery, within-budget delivery, software quality, customer satisfaction etc.) are also considered in related research, we choose labor hours for the following reasons. First, on-time delivery and within-budget delivery measures are deemed less

experience are measured by a marginal decrease/increase in the dependent variable, caused by shared experience variables.

Each project was contracted with a fixed price; so, minimizing labor effort directly increased the profit for the company. Also, because incentives were provided to project members based on the project profit, the project manager was motivated not to incur unnecessary human resource costs during the project.

3.3. Independent Variables

3.3.1. Shared Experience. We include a measure of shared experience (*SharedExp*) used by a number of prior studies [e.g., 4, 12, 15]. Shared experience is computed as follows:

$$SharedExp_{Type} = \frac{\sum_{i=1}^{N-1}\sum_{l=i+1}^{N} w_{ij}}{N(N-1)/2}$$

where *Type* = {*Success, non-Success, Manager, non-Manager*}, *N* is the number of developers in the focal project team, w_{il} is the number of prior projects in which developers *i* and *l* worked together before.

In order to test hypotheses 3, we define shared experience in successful prior projects by counting prior projects of which marginal profits are positive ($SharedExp_{Success}$) and shared experience in non-successful prior projects ($SharedExp_{non\text{-}Success}$) by counting prior projects of which marginal profits are non-positive, respectively. Similarly, in order to test hypothesis 4, we define shared experience of project managers in the focal project team ($SharedExp_{Manager}$) by counting shared experience between project managers and non-managers (including between project managers) and shared experience of non-managers ($SharedExp_{non\text{-}Manager}$) by counting shared experience between non-managers, respectively.

3.3.2. Lapse of Shared Experience. The lapse of a shared experience is defined as the time interval (in months) between the prior experience (i.e., the end date of the prior related project) and the beginning of the focal project. To measure the lapse of shared experience for the project team, we calculate the average lapse across all team members' shared experiences in the project team (*LapseOfSharedExp*). This lapse variable is used to test hypothesis 4.

The following illustration represents how shared experience (*SharedExp*) and its lapse (*LapseOfSharedExp)* are calculated at the project team level.

Suppose that M1 M2, and M3 are project team members. Further, suppose that project members M1 and M3 previously worked together once in project P3 which was completed 9 months ago and project members M2 and M3 worked together twice in projects P1 and P2 which were completed 6 and 3 months ago, respectively. Then shared experience for the focal project team (*SharedExp*) becomes 1 (= (2+1)/(3*2)/2) and the lapse of shared experience becomes 6 (= (9+6+3)/3) months.

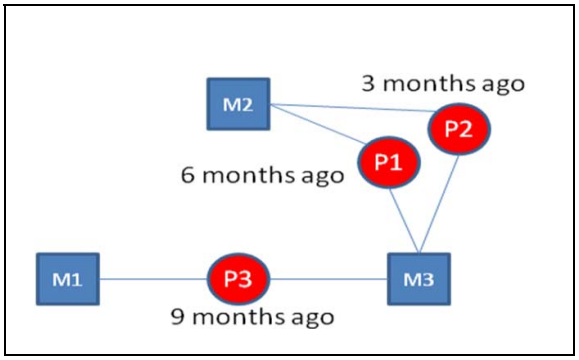

Figure 1. Calculation of SharedExp and *LapseOfSharedExp*

3.4. Control Variables

3.4.1. Estimated Project Cost. We use the project's estimated cost to control for its size (*EstProjectCost*). Since the estimated cost is determined based on the estimated project size, this is an appropriate proxy for the latter.[3] Measures commonly used for project size in prior research include the lines of codes or function points. Unfortunately, the lines of code or function points are not available for many projects at the company we use.

3.4.2. Project Complexity. In order to control for the complexity of the software development project, we use the number of distinct technologies (Num_T) and methodologies (Num_M).

3.4.3. Macroeconomic Condition. We include year dummies for the project end year (*Y2005, Y2006*) to

appropriate because they basically reflect performance based on planned values (i.e., planned cost or schedule) rather than actuals. These planned values are problematic in that they can easily be manipulated at the discretion of project managers or be affected by other factors such as strategic relationships with customers, market competition, experience of planners, and project uncertainty. Second, data on customer satisfaction or system quality is not available to us.

[3] In order to avoid potential endogeneity problems, we use the estimated project cost rather than the actual cost.

control for macroeconomic conditions such as inflation or business cycle, and use 2007 as the baseline for our analyses.

3.4.4. Individual Experience. Prior project experience of individual project members may affect project performance. We measure the project team member's individual experience as the total number of prior projects which he/she experienced before. The average of project team members' individual experience is used for the project team (*IndividualExp*).

3.4.5. Lapse of Individual Experience. Similar to the lapse of shared experience, the lapse of each project experience of project members is calculated, and the average lapse of all project experiences is used for the project team (*LapseOfIndividualExp*).

3.5. Analytical Approach

The most common approach to model the effects of experience is to use a power function. The simple power law formula, $y = cx^{\beta}$, has been widely used in the literature including those in the software development context [4, 13]. The dependent variable y usually represents the cost per unit; the constant c is the initial unit cost; the variable x represents the number of cumulative experience. In order to measure the depreciation effects, we adopt a functional form frequently used in the organizational learning literature as well as economics literature [e.g., 1, 3, 5, 6]: $y = c(x\alpha^t)^{\beta}$. Taking the natural log on both sides produces $\ln(y) = \ln(c) + \beta \times \ln(x) + \beta \times \ln(\alpha) \times t$. Here the variable t represents the time lapse; and α is the depreciation rate parameter. When depreciation effects exist, α takes a value between 0 and 1. When α=1, then there is no depreciation. The basic regression model we use is:

$$\begin{aligned} ln(Effort) = {} & \beta_0 + \beta_1 \times EstProjectCost + \\ & \beta_2 \times Num_m + \beta_3 \times Num_T + \beta_4 \times Year2005 + \\ & \beta_5 \times Year2006 + \beta_6 \times ln(IndividualExp) + \\ & \beta_7 \times ln(SharedExp) + \\ & \beta_8 \times ln(\alpha_1) \times LapseOfIndividualExp + \\ & \beta_9 \times ln(\alpha_2) \times LapseOfSharedExp + \varepsilon \end{aligned}$$

The descriptive statistics and inter-correlations of the variables are shown in Table 1.

4. Results

The results are summarized in Table 2. In the baseline model, we first find that our control variables are significantly associated with project efforts. Larger projects and more complex projects are associated with increased labor effort (*EstProjectCost*: $\beta_1 = 0.174$, $p < 0.01$; Num_M: $\beta_2 = 0.038$, $p < 0.01$; Num_T: $\beta_3 = 0.010$, $p < 0.01$). The year dummies show inflation effects (*Y2005*: $\beta_4 = -0.288$, $p < 0.01$; *Y2006*: $\beta_5 = -0.208$, $p < 0.05$). Finally, individual experience is found to be associated with lower project effort (*IndividualExp*: $\beta_6 = -0.314$, $p < 0.01$) as expected.

Model 1 adds the cumulative shared experience variable to test for hypothesis 1. We observe that the coefficients of all of the control variables remain relatively stable – an indicator that multicollinearity is not an issue. More substantively, we find that the shared experience variable has a significant effect on project performance (i.e., lower project effort; *SharedExp*: $\beta 7 = -0.210$, $p < 0.01$) – hypothesis 1 is supported.

Model 2 adds the lapse variables to test for hypothesis 4. We observe again that the coefficients of all control variables remain relatively stable, and that the time lapse between projects depreciates the effects of individual experience and shared experience (*LapseOfIndividualExp*: $\alpha_1 = 0.999$, $p < 0.1$; *LapseOfSharedExp*: $\alpha_2 = 0.993$, $p < 0.1$) – hypothesis 4 is supported.

Model 3 replaces the total shared experience variable with two separate shared experience variables: One is shared experience in successful prior projects and the other is in non-successful prior projects. We observe that shared experience in successful projects is associated with lowering project effort, while shared experience in non-successful projects is associated with increasing project effort ($SharedExp_{Success}$: $\beta = -0.533$, $p < 0.01$; $SharedExp_{non\text{-}Success}$: $\beta = 0.371$, $p < 0.01$). This means that only shared experience in successful projects increases project performance while shared experience in non-successful experience actually deteriorates performance – hypothesis 2 is supported.

Model 4 has two shared experience variables for project managers and for non-managers, respectively, in order to test for hypothesis 3. The results show that shared experience of project managers is found to lower project effort while shared experience of non-managers does not influence our dependent variable ($SharedExp_{Manager}$: $\beta = -0.472$, $p < 0.01$; $SharedExp_{non\text{-}Manager}$: $\beta = 0.020$, p = ns). – hypothesis 3 is supported.

Table 1. Descriptive Statistics and Inter-Correlations

Variables	Mean	Stdev.	1	2	3	4	5	6	7	8	9	10	11	12	13
1. Effort	19.32	1.59													
2. EstProjectCost	1.51	2.82	0.66***												
3. Num_M	12.22	6.88	0.51***	0.49***											
4. Num_T	43.97	46.80	0.60***	0.62***	0.50***										
5. Year2005	0.38	0.49	-0.02	-0.10**	-0.07*	0.09**									
6. Year2006	0.34	0.48	-0.01	-0.01	0.02	-0.01	-0.56***								
7. ln(IndividualExp)	2.08	0.59	-0.24***	-0.15***	-0.05	-0.26***	-0.12***	0.04							
8. ln(SharedExp)	0.60	0.61	-0.29***	-0.17***	-0.05	-0.23***	-0.34***	0.16***	0.48***						
9. LapseOfIndividualExp	6.93	0.49	0.10**	0.05	0.17***	0.02	-0.19***	0.03	0.52***	0.19***					
10.LapseOfSharedExp	3.79	2.44	0.29***	0.20***	0.25***	0.22***	-0.29***	0.12***	0.02	0.45***	0.13***				
11.ln($SharedExp_{Success}$)	0.31	0.44	-0.33***	-0.18***	-0.09**	-0.24***	-0.31***	0.13***	0.48***	0.81***	0.12***	0.40***			
12.ln($SharedExp_{Non\text{-}Success}$)	0.03	0.11	-0.04	-0.03	-0.06	-0.07	-0.14***	0.11***	0.07	0.23***	-0.04	0.18***	0.13***		
13.ln($SharedExp_{Manager}$)	0.38	0.54	-0.37***	-0.21***	-0.08*	-0.30***	-0.30***	0.12***	0.49***	0.89***	0.19***	0.33***	0.76***	0.23***	
14.ln($SharedExp_{Non\text{-}Manager}$)	0.30	0.40	-0.03	-0.03	0.04	-0.01	-0.22***	0.11**	0.18***	0.65***	0.03	0.34***	0.46***	0.09**	0.28***

Note : Significance Levels: * $p < 0.1$, ** $p < 0.05$, *** $p < 0.01$

Table 2. Regression Results

Variables	Baseline		Model 1		Model 2		Model 3		Model 4	
	Estimate	S.E.	Estimate	S.E.	Estimate	S.E.	Estimate	S.E.	Estimate	S.E.
Intercept	17.805***	0.2150	17.962***	0.2166	17.875***	0.1959	17.702***	0.1901	17.873***	0.1938
EstProjectCost	0.174***	0.0342	0.174***	0.0315	0.175***	0.0172	0.172***	0.0166	0.172***	0.0171
Num_M	0.038***	0.0067	0.033***	0.0063	0.032***	0.0064	0.032***	0.0062	0.032***	0.0064
Num_T	0.010***	0.0016	0.008***	0.0014	0.008***	0.0011	0.007***	0.0011	0.008***	0.0011
Year2005	-0.288***	0.0996	-0.264***	0.1020	-0.243**	0.1059	-0.230**	0.1017	-0.292***	0.1058
Year2006	-0.208**	0.0962	-0.187*	0.0957	-0.178*	0.0954	-0.182**	0.0921	-0.189**	0.0949
ln(IndividualExp)	-0.314***	0.0792	-0.216**	0.0852	-0.271***	0.0924	-0.181**	0.0885	-0.156*	0.0929
ln(SharedExp)			-0.210***	0.0691	-0.189***	0.0759				
LapseOfIndividualExp					0.999*	0.0004	0.999*	0.0005	0.9998	0.0007
LapseOfSharedExp					0.993*	0.0041				
$ln(SharedExp_{Success})$							-0.533***	0.0819		
$ln(SharedExp_{Non\text{-}Success})$							0.371***	0.0958		
$ln(SharedExp_{Manager})$									-0.472***	0.1043
$ln(SharedExp_{Non\text{-}Manager})$									-0.020	0.3376
N	497		497		497		497		497	
R^2	0.679		0.704		0.707		0.728		0.711	

Note : Significance Levels: * $p < 0.1$, ** $p < 0.05$, *** $p < 0.01$

5. Discussion and Conclusion

Table 3 summarizes the testing results for hypotheses. All four hypotheses are supported.

Table 3. Hypotheses Testing Results

Hypotheses	Results
H1. Effects of shared experience	Supported
H2. Effects of shared experience in successful projects > Effects of shared experience in non-successful projects	Supported
H3. Effects of shared experience of project managers > Effects of shared experience of non-managers	Supported
H4. Depreciation effects of shared experience	Supported

We conduct additional analyses to check the robustness of our results. Specifically, we focus on two model-related issues that may potentially challenge our inferences. First, our results may depend on the operationalization of variables. We test an alternative way to represent our shared experience variables in success and non-success projects and those of project managers and non-managers. We define the ratios of shared experience in successful projects and in non-successful projects, respectively (and similarly shared experience of project managers and non-project managers) to entire shared experience, added those ratio variables to Model 2, and test whether these ratio variables have additional effects on the dependent variable. The ratio variables produce the same results as our main model. Second, we test an alternative depreciation model (i.e., a power function of lapse) which has been also widely used in the literature [19]. The main results remain same, too.

Our study has theoretical contribution by showing differential and depreciation effects of shared experience. It also has practical implications on the project staffing problem.

6. References

[1] Argote, L., Beckman, S.L. and Epple, D. 1990. "The persistence and transfer of learning in industrial settings," *Management Science* (36:2), pp 140-154.

[2] Bailey, C.D. 1989. "Forgetting and the learning curve: A laboratory study," *Management Science* (35:3), pp 340-352.

[3] Benkard, C.L. 2000. "Learning and forgetting: The dynamics of aircraft production," *American Economic Review* (90:4), pp 1034-1054.

[4] Boh, W.F., Slaughter, S.A. and Espinosa, J.A. 2007. "Learning from experience in software development: A

multilevel analysis," *Management Science* (53:8), pp 1315-1331.
[5] Boone, T., Ganeshan, R. and Hicks, R.L. 2008. "Learning and knowledge depreciation in professional services," *Management Science* (54:7), pp 1231-1236.
[6] Darr, E.D., Argote, L. and Epple, D. 1995. "The acquisition, transfer, and depreciation of knowledge in service organizations: Productivity in franchises," *Management Science* (41:11), pp 1750-1762.
[7] Ebbinghaus, H. 1885. *Memory: A Contribution to Experimental Psychology*, New York: Dover (Translated by Ruger, H.A. and Bussenius, C.E., 1964).
[8] Epple, D., Argote, L. and Murphy, K. 1996. "An empirical investigation of the microstructure of knowledge acquisition and transfer through learning by doing," *Operations Research* (44:1), pp 77-86.
[9] Faraj, S. and Sproull, L.S. 2000. "Coordinating expertise in software development teams," *Management Science* (46:12), pp 1554-1568.
[10] Globerson, S., Levin, N. and Shtub, A. 1989. "The impact of breaks on forgetting when performing a repetitive task," *IIE Transactions* (21:4), pp 376-381.
[11] Gruenfeld, D.H., Mannix, E.A., Williams, K.Y. and Neale, M.A. 1996. "Group composition and decision making: How member familiarity and information distribution affect process and performance," *Organizational Behavior and Human Decision Processes* (67:1), pp 1-15.
[12] Huckman, R.S., Staats, B.R. and Upton, D.M. 2009. "Team familiarity, role experience, and performance: Evidence from Indian software services," *Management Science* (55:1), pp 85-100.
[13] Narayanan, S., Balasubramanian, S. and Swaminathan, J. 2009. "A matter of balance: Specialization, task variety, and individual learning in a software maintenance environment," *Management Science* (55:11), pp 1861-1876.
[14] Nembhard, D.A. 2000. "The effects of task complexity and experience on learning and forgetting: A field study," *Human Factors* (42:2), pp 272-286.
[15] Reagans, R., Argote, L. and Brooks, D. 2005. "Individual experience and experience working together: Predicting learning rates from knowing who knows what and knowing how to work together," *Management Science* (51:6), pp 869-881.
[16] Standish Group. 2005. *CHAOS Rising: A CHAOS Executive Summary*, West Yarmouth, MA: The Standish Group International.
[17] Thompson, P. 2007. "How much did the Liberty shipbuilders forget?" *Management Science* (53:6), pp 908-918.
[18] Wegner, D.M. 1987. "Transactive memory: A contemporary analysis of the group mind," in *Theories of Group Behavior*, B. Mullen, G.R. Goethals (eds.), New York: Springer-Verlag, pp 185-208.
[19] Wixted, J. T. and Ebbesen E. B. 1991. "On the form of forgetting," *Psychological Science* (2:6), pp 409-41.
[20] Alaranta, M. and Betz, S. 2012. "Knowledge Problems in Corrective Software Maintenance--A Case Study," in *Proc. 45th Hawaii International Conference on System Science (HICSS)*, pp 3746 - 3755.
[21] Kanawattanachai, P. and Yoo, Y. 2007. "The impact of knowledge coordination on virtual team performance over time," *MIS Quarterly* (31:4), pp 782-808.
[22] Qu, G., Ji, S. and Nsakanda, A. 2012. "Project Complexity and Knowledge Transfer in Global Software Outsourcing Project Teams: A Transactive Memory Systems Perspective," in *Proc. 45th Hawaii International Conference on System Science (HICSS)*, pp 3776-3785.
[23] Staats, B.R. 2011. "Unpacking team familiarity: The effects of geographic location and hierarchical role," *Production and Operations Management* (21:3), pp 619-635.