Empowering Leadership, Transactive Memory Systems and Agility in Software Development Teams: A Theoretical Framework

Full Paper

Peng Xu
Department of Management Science and Information Systems
College of Management
University of Massachusetts Boston
Peng.Xu@umb.edu

Yide Shen
Marketing and Business Information Systems Department
Rohrer College of Business
Rowan University
shen@rowan.edu

Abstract

Empowering leadership is crucial in modern software development. However, there is a lack of studies on how empowering leadership affects agility in software development. To fill this gap, we investigate the role of empowering leadership in agility in software development through the lens of transactive memory systems (TMS) theory. In this conceptual paper, we propose a theoretical framework in which TMS plays a mediating role between empowering leadership and agility. This framework advances our understanding of the value of empowering leadership practices in developing TMS, which in turn helps software development teams achieve agility. The proposed leadership practices and their categories also provide guidelines for effectively exercising empowering leadership.

Keywords

Agility, software development teams, empowering leadership, transactive memory systems.

Introduction

Agility is *a firm's ability to move fast to respond to environmental changes and seize novel opportunities* (Dove 1992; Sambamurthy et al. 2003; Trinh 2012). It is considered a significant business capability in today's dynamic business environment. Similarly, agility is also crucial in modern software development (Conboy 2009). Agile software development methodologies have been proposed (Maruping et al. 2009b; Sarker and Sarker 2009) to replace "heavyweight" methodologies that have been criticized for strictly following a project plan and overdoing documents, while failing to respond to changes in requirements and business environment (Lindstrom and Jeffries 2004). The reported benefits of agile methods include increased productivity, faster turnaround, shared learning, and higher developer satisfaction (Lindstrom and Jeffries 2004). Though various agile methodologies differ in practices, tools, and other features, they all share common principles such as an iterative approach, the embrace of changing requirements, frequent delivery, and frequent communications. Initially proposed for small and co-located software development projects, agile methodologies have expanded to other contexts such as large and distributed software development projects (Ramesh et al. 2012; Sarker and Sarker 2009). Increasing evidence indicates that many systems development efforts are attempting to utilize hybrid methods instead of just one method (Vinekar et al. 2006).

Much of the prior research on agile methods focuses on agile practices such as short iteration, pair programming, daily meetings, frequent releases, minimal planning, and working products among others

(Meso and Jain 2006; Yu and Petter 2014). Few have studied the specific role of leadership in agile software development. Leadership is crucial for team effectiveness and agility (Bonner 2010; Carson et al. 2007; Lorinkova et al. 2013; Zaccaro et al. 2001). Leaders can affect project outcomes by influencing the team's objectives, behavior, and culture (Carson et al. 2007). Faraj and Sambamurthy (2006) examined empowering leadership in non-agile software development teams, but failed to find any impact. Tessem (2014) investigated empowering practices used in software development, but not specific leadership behaviors. Other researchers have examined general agile practices, among which leadership was lightly discussed, but it was not the focus (Meso and Jain 2006; Yu and Petter 2014). Bonner (2010) discusses the preferred personality type of a leader in an agile environment.

Different types of leadership have been proposed, such as transformational, directive, empowering, and shared (Bonner 2010; Hoch and Dulebohn 2013; Lorinkova et al. 2013). Of these types, empowering leadership is considered crucial in modern software development (Faraj and Sambamurthy 2006; Tessem 2014). In this study, we investigate the role of empowering leadership in agility in software development from the lens of transactive memory systems (TMS) theory (Wegner 1987), an organizational science theory of group level cognition. Transactive memory systems is a group-level concept that describes the active use of team members' collective knowledge of "who knows what" to collaboratively complete tasks. We argue that empowering leadership can help a team build, maintain, and utilize group TMS, which in turn helps the team achieve agility in software development. In this conceptual paper, we propose a theoretical framework that describes how empowering leadership enables agility via TMS.

Literature Review

Empowering Leadership and Agility in Software Development

Empowering leadership promotes the sharing of power with subordinates in an attempt to raise their level of autonomy (Lorinkova et al. 2013). An empowering leader consults with team members and delegates responsibilities to them. It encourages team members' active participation and self-leadership (Faraj and Sambamurthy 2006). Empowering leadership uses mechanisms to encourage subordinates' specific behaviors, such as collaborative decision-making, active expression of opinions, and supportive teamwork. The factors that support structural empowerment include allowing employees to influence decisions, to gain access to information, and to possess decision-making skills, as well as the extent to which employees are rewarded for participating in these activities (Mills and Ungson 2003; Tessem 2014).

Psychological empowerment concentrates on four cognitions: a) meaningfulness that reflects the employee's perception of the value of the work, b) competence that reflects the employee's belief in his or her own ability to complete the task, c) self-determination that reflects the perception of autonomy at work, and d) impact that reflects the employee's perception of being able to influence the workplace (Tessem 2014). The effectiveness of empowering leadership relies on psychological empowerment, which results from structural empowerment (Tessem 2014). Structural empowerment can be implemented at the beginning of a project, but leaders need time to foster psychological empowerment.

Empowering leadership is considered to be an important leadership style in the knowledge era (Mills and Ungson 2003). Attempts have been made to study its impact in various business contexts, including software development. A longitudinal studies with students observed that teams led by a directive leader exhibit higher initial performance than teams led by an empowering leader, but teams led by an empowering leader exhibit greater improvement (Lorinkova et al. 2013). Magni and Maruping (2013) investigated empowering leadership and improvisation in the retail and finance industries. They found that improvisation is most positively related to performance when empowering leadership is high and overload is low. Empowering leadership, however, is detrimental to the improvisation-performance relationship when team members perceive high degrees of overload. Comparing directive and empowering leadership in traditional software development, Faraj and Sambamurthy (2006) found that the influence of empowering leadership on team performance depends on team experiences and project uncertainty. Strong experiences and high uncertainty call for empowering leadership. When the team has little professional experience, team performance deteriorates under an empowering. Similarly, when the task is relatively certain, empowered teams perform worse.

Tessem (2014) studied team empowerment (not empowering leadership) by examining practices that can help empower team members in both agile and non-agile environments, but he did not focus on leadership's role in empowering team members. Meso and Jain (2006) attempted to understand how agile software development practices can increase a team's adaptability through the lens of complex adaptive systems (CAS) theory. Similarly, the study is concerned with general agile practices, not leadership roles. Yu and Peter (2014) examined agile practices through the lens of a shared mental model. They analyze three main agile practices, stand-up meetings, on-site customers, and system metaphors, and map these practices to shared mental model. However, leadership practices are not discussed in detail. Augustine et al. (2005) also adopted CAS to investigate agile management. Their study proposes adaptive leadership, which includes management practices such as small organic teams, guiding vision, simple rules, championing of open information exchange, and management with a light touch. While we believe adaptive leadership is important, more theoretical examination of the role played by other leadership styles in the context of agility is needed.

In sum, empowering leadership is crucial for knowledge intensive activities such as software development. However, current research has not paid adequate attention to the question of how to leverage empowering leadership to enable agility in software development. Our study aims to answer this question.

Transactive Memory Systems Literature

To understand how empowering leadership behaviors lead to agility in software development teams, we draw on the literature of transactive memory systems (TMS), an organizational science theory of group level cognition that explains how people in collectives rely on each other to obtain expertise in specific domains and to complete tasks collaboratively. Transactive memory refers to the memory existing in a person's mind about what others know and the knowledge that results from it (Wegner 1987). When multiple individuals use their transactive memory to retrieve knowledge from themselves, access knowledge from others, and use the combined knowledge to work toward common goals, a transactive memory system is formed. TMS has been used to study knowledge coordination in dyads, teams, and organizations (e.g., Jackson and Klobas 2008; Lewis 2004; Wegner et al. 1991). When a TMS is formed in a collective, individuals take responsibility for different knowledge domains, which reduces the cognitive load for each member and decreases the amount of redundant knowledge, but still makes a larger pool of knowledge available to all members (Wegner 1987; Wegner 1995). Besides specialized expertise, a well-developed TMS also indicates that individuals place credibility in others' expertise, have a shared understanding of who in the collective knows what, and efficiently coordinate their work. As a result, a well-developed TMS allows members to develop deeper expertise in distinct domains, quickly identify others with expertise relevant to the task at hand, and combine their expertise to work on the task (Kotlarsky and Oshri 2005). In turn, they execute projects effectively and achieve better performance (Austin 2003; Lewis 2004; Lewis 2005; Moreland and Myaskovsky 2000).

As a theory of knowledge coordination, TMS has been used to study group level cognition in various teams, such as new product development (Akgün et al. 2006), knowledge management (Choi et al. 2010), distributed team (Kotlarsky and Oshri 2005; Shen et al. 2016) and information system project teams (Hsu et al. 2011; Lin et al. 2012). However, a limited number of studies have examined TMS in software development teams. Manteli and colleagues (2014) found that different governance decisions (e.g., team configuration, task allocation) have different impacts on TMS in global software development teams. Other researchers have found that TMS plays a major role in software development team performance (Faraj and Sproull 2000) and improves software project technical quality (Maruping et al. 2009a).

These studies shed light on the importance of TMS in helping software development teams achieve better performance. Nevertheless, no research has investigated the effects of TMS on software development agility. We believe examining agility through the lens of TMS theory in this context will advance our understanding of the role TMS plays in software development teams. Transactive memory systems are critical for knowledge intensive teams seeking to achieve high performance (Lewis 2004). A software development team is a typical example of teams that work with knowledge intensive tasks all the time. By using TMS, we provide a theoretical foundation to understand the development of agility in software development teams.

Below, we propose a theoretical framework that examines how empowering leadership behaviors may facilitate TMS development, which in turn increases agility in software development teams.

A Theoretical Framework

We classify six categories of empowering leadership practices: 1) create relational space, 2) encourage collective action, 3) resolve conflicts, 4) develop member capability, 5) foster accountability, and 6) provide feedback. Each category has different impacts on three dimensions of TMS, which in turn enhance agility (Figure 1). Each relationship is discussed below.

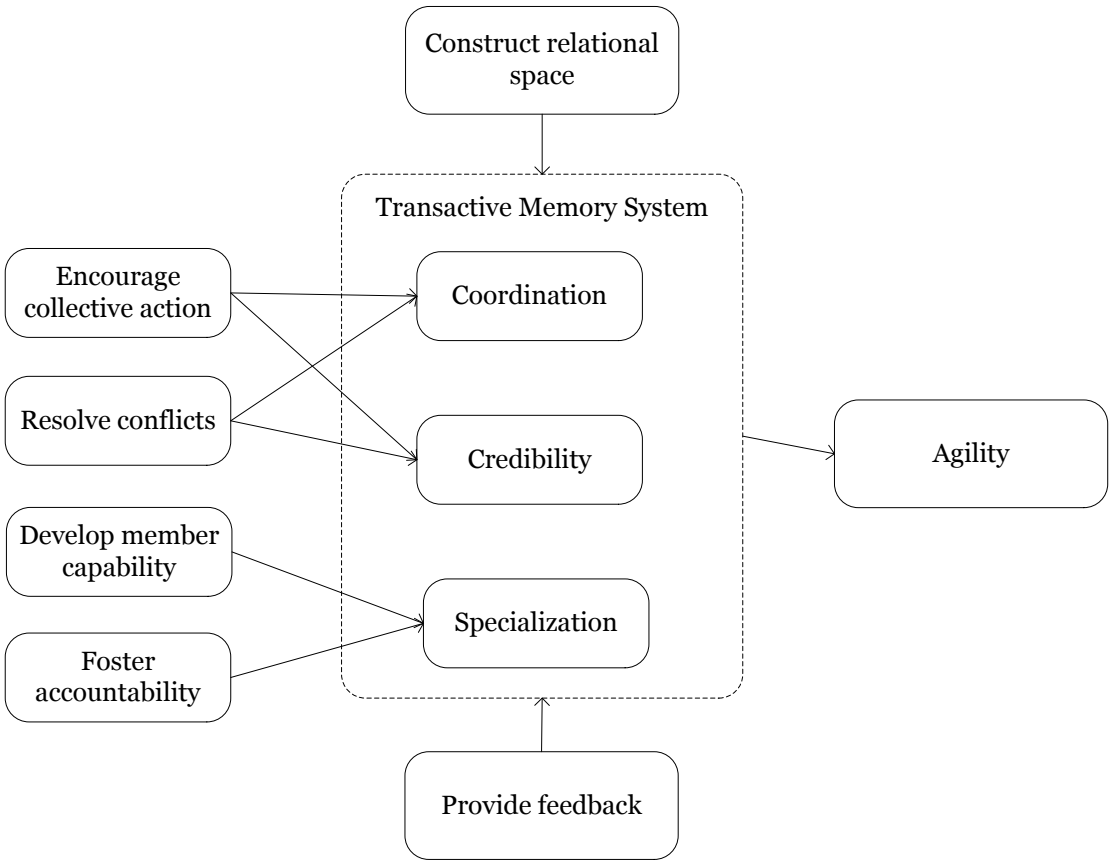

Figure 1. Empowering Leadership, TMS and Agility: A Theoretical Framework

TMS and Agility

Information systems literature has attempted to conceptualize agility in software development (Conboy 2009; Sarker and Sarker 2009). In this research, we adopt Conboy's (2009) definition of agility in software development – "the continual readiness of an ISD method to rapidly or inherently create change, proactively or reactively embrace change, and learn from change while contributing to perceived customer value (economy, quality, and simplicity), through its collective components and relationships with its environment" (p.340). In this section, we discuss how a well-developed TMS facilitates software development teams to achieve agility.

A TMS is a critical mechanism that teams use to leverage each other's diverse expertise (Lewis 2004). Research has identified three indicators of a well-developed TMS in a team (Lewis 2003; Liang et al. 1995; Moreland and Myaskovsky 2000): specialization, credibility and coordination. S*pecialization* refers to the

existence of specialized and differentiated knowledge among team members (Lewis 2003). Specialized knowledge alone is not sufficient to achieve team agility; members need to believe that other members' knowledge is reliable, i.e., *credibility* needs to be present in the team. C*oordination* emphasizes the team's ability to smoothly and efficiently coordinate and combine knowledge with little confusion or misunderstanding (Lewis 2003). The greater the level of these three indicators (or dimensions), the more a TMS is developed and the more valuable it is for efficient knowledge coordination among the team. We argue that the presence of these three dimensions can help software development teams achieve agility.

The *specialization dimension* of TMS enhances agility by providing complementary knowledge in a timely manner when needed. Members' specialized expertise and knowledge complement each other so that the team as a whole can access a larger pool of knowledge relevant to team tasks. Specialized but complementary knowledge can reduce the cognitive burden on each member. Reduced cognitive burden, deep and specialized expertise, and greater task-related, non-redundant knowledge helps software development teams respond rapidly to constant changes, which is essential for team agility.

The *credibility dimension* of TMS positively affects agility by providing a trusting environment and reliable knowledge. When team members believe they can rely on one another for task-critical knowledge and information, they are more likely to utilize others' knowledge and work as a team. Such a trusting environment can help team members create solutions in short periods of time to address changes, which is vital for agility in software development.

The *coordination dimension* of TMS can enhance agility by helping quickly identifying and locating other colleagues with the knowledge relevant to an issue; this provides the ability to quickly contact and enlist the member best suited for an issue (Kotlarsky and Oshri 2005). In other words, TMS can help teams quickly collaborate and respond to new situations and problems, which are essential for team agility.

Thus, the presence of the three dimensions of TMS benefits team agility. In sum, TMS will help a software development team improve its agility.

Empowering Leadership and TMS

In this section, we outline how empowering leadership behaviors, organized into six categories, can help build, maintain, and utilize TMS in software development teams.

Construct Relational Space

One responsibility of empowering leaders is to construct relational space where team members can freely interact and exchange information. Relational space is defined as certain high quality interactions where team members feel safe in the relationship (Lichtenstein and Plowman 2009). Leadership behaviors that fall into this category include setting up an informal meeting infrastructure, guiding information exchange, and promoting the values of information sharing. Prior research has pointed out the importance of information exchange, open discussion, and communication safety in team performance (Augustine et al. 2005; Hirst and Mann 2004; Lindstrom and Jeffries 2004).

Constructing relational spaces can positively affect all three dimensions of TMS: coordination, specialization, and credibility. People factors are extremely important in agile development (Meso and Jain 2006). The role of empowering leadership is more about coaching and motivating than directing (Faraj and Sambamurthy 2006). Instead of telling team members to share information and communicate, an effective empowering leader establishes communication channels and an environment where team members feel comfortable to share and interact. This relational space provides a safe and trusting platform through which team members can form their specific knowledge domains and competence for the project (specialization), understand and build the credibility of one another's specialized expertise (credibility), and start their own coordination (coordination). Therefore, this category of empowering leadership behaviors contributes to all three dimensions of TMS.

Encourage Collective Action

An effective empowering leader is also expected to encourage collective action within a team. Each team member has a role and the team needs to integrate each individual's actions so that each role's performance contributes to collective success (Lichtenstein and Plowman 2009; Zaccaroa et al. 2001). Collective actions require each member to act together based on the project's available resources, task requirements, and member characteristics (Zaccaroa et al. 2001). Leaders can encourage collective action by promoting values of collaborative decision-making and teamwork. Leaders also need to foster collective ownership so that all team members are responsible for product quality. For example, through daily and iteration planning meetings, leaders can engage all team members, solicit suggestions, invite members to contribute to the planning process, consult them on decisions, publicly recognize the team's efforts, and build the value of teamwork into a rewarding system and team culture.

Collective actions such as collective decision-making, information processing, and ownership are crucial in agile development (Tessem 2014; Zaccaroa et al. 2001). By encouraging collective action, leaders can motivate the team to coordinate their work, which will benefit the coordination dimension of TMS. In addition, it can also help build an open, friendly environment that develops high levels of credibility among members. Therefore, empowering leadership practices in this category can positively affect the coordination and credibility dimensions of TMS.

Resolve Conflict

Empowering leaders should detect and surface conflicts, and encourage the team to resolve conflicts among themselves. Task conflict refers to disagreements among group members about decisions, viewpoints, ideas, and opinions (Langfred 2007). Empowered team members actively participate in software development activities such as making important decisions on feature selection, design strategies, and iteration planning. Given that change occurs frequently in agile environments, new decisions and solutions are needed constantly (Augustine et al. 2005). It is inevitable that multiple voices and opinions emerge, leading to conflicts. Conflicts also arise when team members compete for resources and tasks. In this circumstance, an effective empowering leader surfaces the conflict, helps the team acknowledge it, and provides an opportunity to discuss solutions (Lichtenstein and Plowman 2009; Uhl-Bien et al. 2007).

An empowering leader encourages the team to resolve conflicts on its own. By doing so, team members can explore and clarify the relationships between tasks and between a task and people, thus updating their memory of "who knows what". This helps them coordinate more effectively among themselves. Thus, it can positively affect the coordination dimension of TMS. By empowering team members to solve their conflicts, it also gives them the opportunity to judge the credibility and quality of others' knowledge, thus contributing to the credibility dimension. Therefore, empowering leadership practices in this category can positively affect the coordination and credibility dimensions.

Develop Member Capability

This category focuses on helping team members understand the project and its environment, and thus develop project-specific knowledge and skills. The practices include inspiring members to work through difficulties, helping them develop capabilities to collect and process project information, encouraging them to take initiative, and embracing uncertainty.

These practices can help team members build specialized knowledge/competence, i.e., the specialization dimension of TMS. By working through difficulties and developing the capabilities to process and structure project-related information, team members learn their tasks better and build their project-specific knowledge (Nan and Kumar 2013). Through experiential learning, team members develop new knowledge and the competence required (Peng et al. 2013). Uncertainty in today's business environment is inevitable, especially in agile development. Changes in business environment and customer requirements need to be processed effectively by the development team to achieve agility. Helping team members embrace uncertainty helps them develop capabilities and knowledge for the volatile nature of agile development. Motivating team members to take initiative for changes can also help them recognize their strengths and position themselves within the team. This further helps them develop their specialized

knowledge and competence for the project. Thus, empowering practices in this category can positively affect the specialization dimension.

Foster Accountability

Empowering leaders delegate authority and support the team's autonomy, which is defined as the amount of freedom and discretion an individual has to carry out assigned tasks (Langfred 2007). Team autonomy is crucial for a team to respond to change (Maruping et al. 2009b). More autonomy means more responsibility and more competence.

Through autonomy, empowering leaders motivate their teams to develop and advance projects and team knowledge and competence. Team members take charge of various tasks. Such a process helps team members specialize in various knowledge domains of development efforts and enable them to act quickly when change happens. This positively influences the specialization dimension of TMS.

Provide Feedback

Change is routine in agile development. Unlike traditional software development projects where requirements are finalized upfront and feedback is provided toward the end, agile development needs to be both reactive and proactive to frequent changes. These changes introduce adjustments in various aspects of the project such as tasks, expertise required, and task assignments. An empowering leader needs to facilitate role adjustments and provide feedback when changes happen (Zaccaroa et al. 2001). Throughout the project, the leader needs to assess the effectiveness of psychological empowerment and update the empowerment structure accordingly. When changes happen, but are not accepted by the team, the leader needs to provoke a state of dis-equilibrium and amplify change so that the team is aware of the changes and is willing to address them (Lichtenstein and Plowman 2009). The leader guides recombination and re-planning, if necessary. During dynamic changes, the leader also needs to ensure the constraints of the project are applied to these changes.

By providing feedback, leaders help members re-evaluate and clarify the links among tasks, knowledge needed for the tasks, and the people who possess the needed knowledge. It helps team members further develop their competence specialization for the project. All these feedback activities also provide opportunities for team members to adjust their behaviors to embrace change and coordinate more effectively. It also allows members to further establish and build their credibility. Thus, feedback – one category of empowering leadership practices — contributes to all three dimensions of TMS.

Table 1 summarizes the six categories of practices and the TMS dimensions they influence.

Category	Empowering Leadership Practices	Influenced Dimension(s)
Construct relational space	• Set up informal meeting infrastructure • Guide information exchange • Promote values of information sharing	• Specialization • Credibility • Coordination
Encourage collective action	• Promote values of collaborative decision making • Promote and reward team work • Foster collective ownership	• Coordination
Develop member capability	• Inspire members to work through difficulties • Encourage members to develop capabilities to collect and process project information • Encourage members to take initiative • Encourage experiential learning • Encourage the embracing of uncertainty	• Specialization
Resolve conflicts	• Surface conflicts (generate discussion and solutions) • Encourage conflict resolution among team members	• Credibility • Coordination

Foster accountability	• Encourage and support team member autonomy and responsibility • Delegate authority	• Specialization
Provide feedback	• Facilitate role adjustments when needed • Assess psychological empowerment • Monitor and update empowerment structure • Detect changes • Provoke state of dis-equilibrium • Amplify change within the team • Lead/facilitate recombination/regrouping/re-planning • Control emergent changes • Apply constraints to its rapid change • Drive collaboration through shared terminology & symbols	• Specialization • Credibility • Coordination

Table 1. Empowering Leadership Practices

Discussion and Conclusion

In this conceptual paper, we examine the role of empowering leadership in agile development. In particular, drawing from TMS theory, we study how empowering leadership helps software development teams develop TMS, which in turn positively affects team agility. Our study makes several contributions. First, it contributes to the agility literature by highlighting the impact of empowering leadership. Prior research on agile software development concentrated on proposed agile practices from industry (Drury et al. 2012; Lindstrom and Jeffries 2004; Meso and Jain 2006; Ramesh et al. 2012), agile practices (not leadership) that can empower team members (Tessem 2014), or empowering leadership and directive leadership in non-agile contexts (Faraj and Sambamurthy 2006) – not how empowering leadership contributes to agility. Our study contributes to this literature by focusing on empowering leadership and its enabling role in creating an environment where agility can happen. Second, it contributes to the TMS literature, especially in the context of software development. Prior research has recognized the impact of TMS in knowledge intensive teams seeking to achieve high performance (Lewis 2004). Software development is a knowledge intensive process and can definitely benefit from well-developed TMS. However, limited research has been performed to link TMS to software development projects and to agility, in particular. Our study fills this gap. Our model provides a theoretical framework in which TMS plays a mediating role between empowering leadership and agility. Third, this paper contributes to practices. The proposed framework presents six categories of empowering leadership practices that managers can use as behavior guidelines. These guidelines help team leaders effectively exercise empowering leadership, facilitate TMS development, and achieve agility in software development teams.

References

Akgün, A. E., Byrne, J. C., Keskin, H., and Lynn, G. S. 2006. "Transactive Memory System in New Product Development Teams," *IEEE Transactions on Engineering Management* (53:1), pp. 95-111.

Augustine, S., Payne, B., Sencindiver, F., and Woodcock, S. 2005. "Agile Project Management: Steering from the Edges," *Communications of the ACM* (48:12), pp. 85-89.

Austin, J. R. 2003. "Transactive Memory in Organizational Groups: The Effects of Content, Consensus, Specialization, and Accuracy on Group Performance," *Journal of Applied Psychology* (88:5), pp. 866-878.

Bonner, N. A. 2010. "Predicting Leadership Success in Agile Environments: An Inquiring Systems Approach," *Academy of Information and Management Sciences Journal* (13:2), pp. 83-103.

Carson, J. B., Tesluk, P. E., and Marrone, J. A. 2007. "Shared Leadership in Teams: An Investigation of Antecedent Conditions and Performance," *Academy of Management Journal* (50:5), pp. 1217–1234.

Choi, S. Y., Lee, H., and Yoo, Y. 2010. "The Impact of Information Technology and Transactive Memory Systems on Knowledge Sharing, Application, and Team Performance: A Field Study," *MIS Quarterly* (34:4), pp. 855-870.

Conboy, K. 2009. "Agility from First Principles: Reconstructing the Concept of Agility in Information Systems Development," *Information Systems Research* (20:3), pp. 329–354.
Dove, R. 1992. "What Is All This Talk About Agility - the 21st Century Manufacturing Enterprise Strategy," Japan Management Association Research.
Drury, M., Conboy, K., and Power, K. 2012. "Obstacles to Decision Making in Agile Software Development Teams," *Journal of Systems and Software* (85:6), pp. 1239-1254
Faraj, S., and Sambamurthy, V. 2006. "Leadership of Information Systems Development Projects," *IEEE Transactions on Engineering Management* (53:2), pp. 238-249.
Faraj, S., and Sproull, L. 2000. "Coordinating Expertise in Software Development Teams," *Management Science* (46:12), pp. 1554-1568.
Hirst, G., and Mann, L. 2004. "A Model of R&D Leadership and Team Communication: The Relationship with Project Performance," *R&D Management* (32:2), pp. 147-160.
Hoch, J. E., and Dulebohn, J. H. 2013. "Shared Leadership in Enterprise Resource Planning and Human Resource Management System Implementation," *Human Resource Management,* (23:1), pp. 114–125.
Hsu, J. S. C., Shih, S. P., Chiang, J. C., and Liu, J. Y. C. 2011. "The Impact of Transactive Memory Systems on Is Development Teams' Coordination, Communication, and Performance," *International Journal of Project Management* (30:3), pp. 329-340.
Jackson, P., and Klobas, J. 2008. "Transactive Memory Systems in Organizations: Implications for Knowledge Directories," *Decision Support Systems* (44:2), pp. 409-424.
Kotlarsky, J., and Oshri, I. 2005. "Social Ties, Knowledge Sharing and Successful Collaboration in Globally Distributed System Development Projects," *European Journal of Information Systems* (14:1), pp. 37-48.
Langfred, C. W. 2007. "The Downside of Self-Management: A Longitudinal Study of the Effects of Conflict on Trust, Autonomy, and Task Interdependence in Self-Managing Teams," *Academy of Management Journal* (50:4), pp. 885–900.
Lewis, K. 2003. "Measuring Transactive Memory Systems in the Field: Scale Development and Validation," *Journal of Applied Psychology* (88:4), pp. 587–604.
Lewis, K. 2004. "Knowledge and Performance in Knowledge-Worker Teams: A Longitudinal Study of Transactive Memory Systems," *Management Science* (50:11), pp. 1519-1533.
Lewis, K. 2005. "Transactive Memory Systems, Learning and Learning Transfer," *Organization Science* (16:6), pp. 581-598.
Liang, D. W., Moreland, R. L., and Argote, L. 1995. "Group Versus Individual Training and Group Performance: The Mediating Role of Transactive Memory " *Personality and Social Psychology Bulletin* (21), pp. 384-393.
Lichtenstein, B. B., and Plowman, D. A. 2009. "The Leadership of Emergence: A Complex Systems Leadership Theory of Emergence at Successive Organizational Levels," *The Leadership Quarterly* (20), pp. 617–630.
Lin, T.-C., Hsu, J. S.-C., Cheng, K.-T., and Wu, S. 2012. "Understanding the Role of Behavioural Integration in Isd Teams: An Extension of Transactive Memory Systems Concept," *Information Systems Journal* (22:3), pp. 211-234.
Lindstrom, L., and Jeffries, R. 2004. "Extreme Programming and Agile Software Development Methodologies," *Information Systems Management* (21:3), pp. 41-52.
Lorinkova, N. M., Pearsall, M. J., and Jr., H. P. S. 2013. "Examining the Differential Longitudinal Performance of Directive Versus Empowering Leadership in Teams," *Academy of Management Journal* (56:2), pp. 573-596.
Magni, M., and Maruping, L. M. 2013. "Sink or Swim: Empowering Leadership and Overload in Teams' Ability to Deal with the Unexpected," *Human Resource Management,* (52:5), pp. 715–739.
Manteli, C., van den Hooff, B., and van Vliet, H. 2014. "The Effect of Governance on Global Software Development: An Empirical Research in Transactive Memory Systems," *Information and Software Technology* (56:10), pp. 1309-1321.
Maruping, L., Zhang, X., and Venkatesh, V. 2009a. "Role of Collective Ownership and Coding Standards in Coordinating Expertise in Software Project Teams," *European Journal of Information Systems* (18:4), p. 355.
Maruping, L. M., Venkatesh, V., and Agarwal, R. 2009b. "A Control Theory Perspective on Agile Methodology Use and Changing User Requirements," *Information Systems Research* (20:3), pp. 377–399.

Meso, P., and Jain, R. 2006. "Agile Software Development: Adaptive Systems Principles and Best Practices," *Information Systems Management* (23:3), pp. 19-30.
Mills, P. K., and Ungson, G. R. 2003. "Reassessing the Limits of Structural Empowerment: Organizational Constitution and Trust as Controls," *Acadamy Management Review* (28:1), pp. 143–153.
Moreland, R. L., and Myaskovsky, L. 2000. "Exploring the Performance Benefits of Group Training: Transactive Memory or Improved Communication?," *Organization Behavior and Human Decision Processes* (82:1), pp. 117-133.
Nan, N., and Kumar, S. 2013. "Joint Effect of Team Structure and Software Architecture in Open Source Software Development," *IEEE Transactions on Engineering Management* (60:3), pp. 592-603.
Peng, G., Mu, J., and Benedetto, C. A. D. 2013. "Learning and Open Source Software License Choice," *Decision Sciences* (44:4).
Ramesh, B., Mohan, K., and Cao, L. 2012. "Ambidexterity in Agile Distributed Development: An Empirical Investigation," *Information Systems Research* (23:2), pp. 323-339.
Sambamurthy, V., Bharadwaj, A., and Grover, V. 2003. "Shaping Agility through Digital Options: Reconceptualizing the Role of Information Technology in Contemporary Firms," *MIS Quarterly* (27:2), pp. 237-263.
Sarker, S., Munson, C. L., Sarker, S., and Chakraborty, S. 2009. "Assessing the Relative Contribution of the Facets of Agility to Distributed Systems Development Success: An Analytic Hierarchy Process Approach," *European Journal of Information Systems* (18:4), pp. 285–299.
Sarker, S., and Sarker, S. 2009. "Exploring Agility in Distributed Information Systems Development Teams: An Interpretive Study in an Offshoring Context " *Information Systems Research* (20:3), pp. 440–461.
Shen, Y., Gallivan, M., and Tang, X. 2016. "The Impact of Subgroup Formation on Transactive Memory Systems and Performance in Distributed Teams," *International Journal of e-Collaboration* (12:1), pp. 44-66.
Tessem, B. 2014. "Individual Empowerment of Agile and Non-Agile Software Developers in Small Teams," *Information and Software Technology* (56:8), pp. 873–889.
Trinh, T. P. 2012. "Enterprise Systems and Organizational Agility: A Review of the Literature and Conceptual Framework," *Communications of the Association for Information Systems* (31:Article 8), pp. 167-193.
Uhl-Bien, M., Marion, R., and McKelvey, B. 2007. "Complexity Leadership Theory: Shifting Leadership from the Industrial Age to the Knowledge Era," *The Leadership Quarterly* (18), pp. 298–318.
Vinekar, V., Slinkman, C. W., and Nerur, S. 2006. "Can Agile and Traditional Systems Development Approaches Coexist? An Ambidextrous View," *Information Systems Management* (23:3), pp. 31-42.
Wegner, D. M. 1987. "Transactive Memory: A Contemporary Analysis of the Group Mind," in *Theories of Group Behavior,* B. Mullen and G.R. Goethals (eds.). New York: Springer-Verlag, pp. 185-208.
Wegner, D. M. 1995. "A Computer Network Model of Human Transactive Memory," *Social Cognition* (13:3), pp. 319-339.
Wegner, D. M., Erber, R., and Raymond, P. 1991. "Transactive Memory in Close Relationship," *Journal of Personality and Social Psychology* (61:6), pp. 923-929.
Yu, X., and Petter, S. 2014. "Understanding Agile Software Development Practices Using Shared Mental Models Theory," *Information and Software Technology* (56), pp. 911–921.
Zaccaro, S. J., Rittman, A. L., and Marks, M. A. 2001. "Team Leadership," *Leadership Quarterly* (12:4), pp. 451–483.