Leading Agile Teams: An Exploratory Study of Leadership Styles in Agile Software Development

Emergent Research Forum Papers

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

Leadership is crucial for agile team effectiveness and performance. The roles played by project leaders have significantly changed in agile methodologies, but very limited studies on leadership have been conducted in the agile software development context. Motivated by this gap, this study aims to investigate the roles of various leadership styles in agile methodologies and understand how different types of leadership can collaboratively improve an agile project’s performance. We will conduct an exploratory case study to develop a research model. Our study will contribute to the theoretical understanding of leadership roles in agile projects. The findings will also provide guidance to project management practices.

Keywords

Agile software development, agile teams, leadership styles, leadership in agile methods.

Introduction

Over the years, software development methodologies have evolved from ad hoc approaches to a proliferation of methodologies. Traditional “heavyweight” methodologies emphasize pre-planning, tight control, extensive documentation, and detailed design (Ramsin and Paige 2008). A new set of “lightweight” methodologies, called agile methodologies, has quickly become popular in the past 10 years. Though various agile methodologies differ in practices, tools, and many other features, they all share common principles such as an iterative approach, the embrace of changing requirements, frequent delivery, and frequent communications. Different from traditional “heavyweight” methodologies, agile methodologies rely on dynamic adaptation and lean processes (Maruping et al. 2009; Sarker and Sarker 2009).

Efforts have been made to study how to adopt and tailor agile methodologies for various projects such as large software development projects and distributed projects (Ramesh et al. 2012; Sarker and Sarker 2009). However, little has been done to study the role of leadership in agile methodologies. Leadership is crucial for team effectiveness (Carson et al. 2007; Lorinkova et al. 2013). It can influence a project’s objectives, motivate desired behaviors in pursuit of these objectives, change business processes, and affect group culture (Carson et al. 2007). However, we know surprisingly little about leaders’ roles in agile projects and how different leadership styles affect agile projects’ performance. A conceptual framework was proposed on the relationship between vertical leadership and shared leadership in ERP implementations (Hoch and Dulebohn 2013). Unfortunately, no empirical evidence is available to verify the framework. Faraj and Sambamurthy (Faraj and Sambamurthy 2006) examined empowering
leadership in software development using non-agile methodologies. Tessem (2014) investigated empowering practices in both agile and non-agile environments. This study, however, focuses more on different empowering practices, less on overall leadership roles.

Self-organizing teams still need guidance from a leader (Druskat & Wheeler, 2004). Though agile methods propose collaborative, self-directed teams, projects face challenges such as conflicting priorities for decisions and members’ unwillingness to take ownership, participate, or commit to a decision (Tessem, 2014). The roles played by project managers and team members have significantly changed in agile methodologies, but very few studies on leadership have been conducted in the context of agile software development. Motivated by such a gap, this study aims to investigate the roles of various leadership styles in agile methodologies and understand how different leadership styles can improve an agile project’s performance.

**Literature Review**

**Agile Methodology**

“Heavyweight” methodologies such as the waterfall model have been criticized for strictly following the project plan and overdoing documents, but failing to respond to changes in requirements and business environment (Lindstrom and Jeffries 2004). To address these limitations, agile methodologies such as eXtreme Programming and Scrum have been proposed and adopted (Conboy 2009; Vinekar et al. 2006). Agile methodologies propose to execute multiple short iterations, each of which focuses on prioritized features and delivers working products (Lindstrom and Jeffries 2004). Customers are invited to work closely with the development team and provide frequent feedback. Instead of documenting all important design and development decisions, agile teams are encouraged to use oral communications to share knowledge and update project status. The reported benefits of agile methods include increased productivity, faster turnaround, shared learning, and higher developer satisfaction (Lindstrom and Jeffries 2004). Agile methodologies have also been used for large and distributed software development projects (Ramesh et al. 2012; Sarker and Sarker 2009). More evidence indicates that many systems development efforts are attempting to utilize hybrid methods instead of just one method (Vinekar et al. 2006).

**Leadership Research in Software Development**

There are very few studies on leadership behaviors in software development. These studies lack both theoretical and empirical support.

Obstacles to decision making in agile software development teams have motivated the investigation of empowerment (Drury et al. 2012; Tessem 2014). Tessem (2014) compared empowerment practices in agile and non-agile environments. Though this study describes the role played by project managers in empowering processes, it focuses on individual team members and their perceptions of empowerment. It does not directly address the change in leadership roles in agile methodologies.

Faraj and Sambamurthy (2006) compared directive leadership and empowering leadership in teams that use traditional software development methodologies. Directive leaders make decisions, integrate individual output, and manage team boundaries. On the other hand, empowering leaders consult team members and makes joint decisions with them. The findings suggest that experienced teams and high project uncertainty call for empowering leadership, but empowered teams perform worse than the ones adopting directive leadership when teams have low levels of experience and/or tasks are certain.

In addition, shared leadership has been examined in ERP projects that used traditional software development methodologies (Hoch and Dulebohn 2013). In shared leadership, multiple team members engage in leadership roles, instead of having only one hierarchical leader in a team (Hoch and Dulebohn 2013). Hoch and Dulebohn (2013) proposed a conceptual framework to explain how to create an environment that is conducive for the development of shared leadership. However, no empirical evidence is available.
Many have agreed that leadership in agile environments is significantly different from that in traditional software development (Bonner 2010). However, very limited research has been conducted on the subject. More studies are needed in this field.

**Research Framework**

Several leadership styles have been proposed by management literature such as empowering leadership (Lorinkova et al. 2013), shared leadership (Carson et al. 2007), self-leadership (Neck and Houghton 2006), and directive leadership (Lorinkova et al. 2013). Multiple leadership styles are not mutually exclusive to one another and can co-exist in a team (Hoch and Dulebohn 2013). In this study, we will explore these leadership styles and their roles in agile projects.

**Empowering Leadership**

Empowering leadership promotes the sharing of power with subordinates in an attempt to raise their level of autonomy (Lorinkova et al. 2013). It can be achieved via both structural empowerment and psychological empowerment (Tessem 2014). Structural empowerment uses mechanisms to encourage subordinates’ specific behaviors such as collaborative decision-making, actively expressing opinions, and supporting team work, while psychological empowerment concentrates on employees’ perceptions of their value, competence, autonomy, and impact (Tessem 2014). The effect of empowering leadership relies on psychological empowerment, which is an effect of structural empowerment, combined with other factors such as individual characteristics, work design, and organizational support (Tessem 2014). Structural empowerment can be implemented at the beginning of a project, but leaders need time to foster psychological empowerment.

The underlying principle of empowering leadership is consistent with that of agile methodologies. Agile methods promote shared responsibility among team members. Team members are involved in decision-making processes and even make decisions on their own. Instead of waiting for assignment and command, team members can choose their own tasks and even plan their schedule. These agile practices encourage voluntary, proactive participation of all team members and provide structural empowerment mechanisms that can empower team members.

Empowering leadership faces challenges in agile projects. First, it is not uncommon to have passive followers on the team who are unwilling to be proactive (Conboy 2009). Team members who are inexperienced with a project or agile methods may also be reluctant to play empowered roles. Second, a development team has to go through several phases to complete a project (Lorinkova et al. 2013). In the early phases of a project, empowering practices may not take effect immediately.

**Shared Leadership**

Instead of having a single hierarchical leader responsible for a team, shared leadership promotes multiple team members engaging in various forms of leadership (Hoch and Dulebohn 2013). Team members lead each other toward achieving project goals. Shared leadership is possible among team members even when there also is a single designated leader (Hoch and Dulebohn 2013).

The principles of shared leadership are similar to those of empowering leadership, but take a more aggressive approach in encouraging team member participation. In empowering leadership, though team members are encouraged to share more responsibility, the project manager is still the top authority. Empowered team members may not lead the project. In shared leadership, team members not only share more responsibility, but also share more authority from designated project leaders. Therefore, these two concepts are different (Carson et al. 2007). The multiple leader roles can be formally designated or informally recognized within the team; the length of leader roles can be either short-term or long-term; and the scope of the leader roles can have the exact same scope as the project or only focus on part of the project.
Shared leadership’s principles are also consistent with agile principles. Shared leadership enables capable team members to take on more responsibility and be more proactive in projects. When multiple team leaders share authority, the structure of the team becomes flatter. One designated project leader is no longer a bottleneck for information collection and processing. Such a flat team structure can accelerate faster decision-making and actions, which is the spirit of agile methods.

Shared leadership also faces challenges in agile projects. One challenge is how to coordinate among multiple leaders. Multiple authorities can become a source of conflict. Another challenge is the transition among leaders. In shared leadership, leader roles may be assumed by multiple team members for various lengths of time. Because they are not necessarily formally designated roles, the transitions between different leaders can pose challenges.

**Self-leadership**

Self-leadership is a form of leadership where individuals control their own behaviors and lead themselves to achieve desired goals (Neck and Houghton 2006, p. 270). Self-leadership incorporates a broad range of behavior-focused strategies such as self-regulation, self-control, self-management and cognitive-oriented strategies such as intrinsic motivation (Houghton and Neck 2002). Self-leadership believes that all members are capable of leading themselves to some extent (Pearce and Manz 2005).

In agile projects, team members are expected to take more initiative and responsibility in choosing their own tasks, planning their schedules, and participating in team decision-making. Such expectations require team members to exercise self-leadership and lead themselves to achieve desired goals. Although agile practices encouraging self-organizing teams, research has shown that such teams still need leaders that facilitate self-leadership strategies (Manz and Sims 1987; Neck et al. 1996).

The challenge of self-leadership in agile methodologies is “individual leadership” where members do not communicate with others and do not inform others about changes made (Moe et al. 2009).

**Directive Leadership**

Directive leadership aims at “actively structuring subordinates’ work through providing clear directions and expectations regarding compliance with instructions” (Lorinkova et al. 2013, p. 537). Such leadership requires project managers to direct team members’ activities; estimate costs, time, and schedules; assign roles and tasks; monitor and control the project; act as liaison to the client and upper management; and control changes (Bonner 2010; Klein et al. 2006). Directive leadership plays a major role in heavyweight methodologies. Though the principle of directive leadership is not compatible with that of agile methodologies, its role in agile methodologies is worth investigating given that many agile methodologies are not adopted as what they are, but are tailored for various development contexts.

**The Dynamic Interactions among Leadership Styles**

A development team has to go through several phases to complete a project. The phases include the team formation and task compilation phase where team members join the team and get familiar with the project; the role compilation phase where team members adjust to their roles and leadership styles; and the team compilation phase where team members develop shared understanding of their routines (Lorinkova et al. 2013).

Prior literature suggests that multiple leadership styles can co-exist and can influence one another in a project. For example, Hoch and Dulebohn (2013) propose that empowering leadership can help build shared leadership in ERP projects. We believe empowering leadership, shared leadership, self-leadership, and directive leadership all play some role in projects using agile methodologies. Their impact on project performance may vary in different phases. The co-existence of multiple leadership styles and their interactions are crucial to the success of an agile project.
Therefore, in this study, we will investigate how these leadership styles are used in agile software development and how they can affect an agile project’s performance in terms of product quality and process efficiency. We will also examine how different leadership styles affect each other in agile development. Figure 1 summarizes the research framework.

**Research Method**

The goal of our study is to build a research model that can explain how different leadership styles affect agile project performance. The nature of our study is exploratory, not confirmatory. We believe a qualitative study method is suitable for this purpose (Yin 2015). We will adopt the grounded theory approach to collect and analyze our data (Corbin and Strauss 2007). The lack of a systematic, theoretical research framework on leadership in agile projects motivates the use of the qualitative approach to inductively develop a research model.

We will adopt the case study approach. The unit of analysis is an agile project. We will get access to multiple agile projects. Multiple projects offer deeper understanding of leadership roles and improve the confidence of our findings (Miles et al. 2013).

Qualitative data will be collected via on-site interviews, observations, and review of documentation, if applicable. Data collection and analysis will be conducted in an iterative fashion (Corbin and Strauss 2007). Key contact persons will help us identify projects and teams. We will conduct semi-structured interviews with project leaders and members.

All interviews will be recorded and transcribed. We will first conduct within-case analysis for each case and then do cross-case analysis to compare the cases. Two researchers will separately analyze the qualitative data and develop coding schema. The two coding schemes will be compared and discussed. The final coding scheme agreed upon by both researchers will be used. We are refining our interview questions and strategies. We will soon contact our potential research sites.

**Contributions and Conclusion**

We believe our study will make contributions to both academia and practice. First, our research will make contributions to the theoretical understanding of leadership roles in agile projects. As one family of the most important software development methodologies, agile methodologies have drawn attention from academia. A vast amount of work has been done to understand the concept of agility in system development (Conboy 2009), obstacles in agile projects (Drury et al. 2012), and the adoption of agile methodologies in various types of projects and environments (Lindstrom and Jeffries 2004; Ramesh et al. 2012). However, little has been done to fully understand leadership roles in teams using agile
Leadership Styles in Agile Software Development

methodologies. Our study aims to build a research framework that will explain how various leadership types can collaboratively improve agility and performance. Second, the results of our study will provide practical guidance for agile teams and management to effectively lead agile projects and coordinate efforts.

References


