Collaboration in Agile Software Development: Concept and Dimensions

Dinesh Batra  
*Florida International University*, batra@fiu.edu

Weidong Xia  
*Florida International University*

Mingyu Zhang  
*Florida International University*

Follow this and additional works at: [https://aisel.aisnet.org/cais](https://aisel.aisnet.org/cais)

Recommended Citation
DOI: 10.17705/1CAIS.04120  
Available at: [https://aisel.aisnet.org/cais/vol41/iss1/20](https://aisel.aisnet.org/cais/vol41/iss1/20)

This material is brought to you by the AIS Journals at AIS Electronic Library (AISeL). It has been accepted for inclusion in Communications of the Association for Information Systems by an authorized administrator of AIS Electronic Library (AISeL). For more information, please contact elibrary@aisnet.org.
Collaboration in Agile Software Development: Concept and Dimensions

Dinesh Batra  
Florida International University  
batra@fiu.edu

Weidong Xia  
Florida International University

Mingyu Zhang  
Florida International University

Abstract:

One of the four values listed in the Agile Manifesto emphasizes customer collaboration over contract negotiation, yet the literature has not explained what constitutes customer collaboration and how to assess it. Little research has examined the nature and dimensions of collaboration in the context of agile software development. Based on a grounded theory methodology and using interview data collected from five software development outsourcing vendors in China, we explore the nature and key underlying dimensions of collaboration in agile software development. Five major dimensions of collaboration emerged from our analysis: mutual benefits, engagement, coordination, communication, and knowledge sharing. In turn, each dimension comprises key subdimensions that provide a comprehensive view of collaboration. By revealing the underlying nature and key dimensions, we provide a conceptual basis for operationalizing collaboration that one can employ in future quantitative studies on agility and other project outcomes. Our study results suggest that collaboration in agile software development is multifaceted and mutually occurring in both directions between the customer and the vendor rather than single dimensional as the term “customer collaboration” in the Agile Manifesto indicates.

Keywords: Collaboration, Coordination, Agile Software Development, Communication.

This manuscript underwent peer review. It was received 09/23/2016 and was with the authors for 6 months for 2 revisions. Indranil Bose served as Associate Editor.
1 Introduction

One of the four values listed in the Agile Manifesto emphasizes customer collaboration over contract negotiation, which inherently links collaboration and agility (Highsmith, 2002). In particular, agile practices purport to enhance collaboration between IT and business teams (Yu & Petter, 2014). However, the Agile Manifesto uses the term “customer collaboration”, which focuses on vendors’ activities to cause customers to engage more with the software development process. Because organizations now use agile practices in both small projects with co-located teams and large, distributed projects, we need to extend collaboration to reflect agile software development’s evolving nature. The close relationship between the Chinese software development vendors and the Japanese clients provides an ideal setting for conducting research to identify the key dimensions of collaboration.

The notion of collaboration is not new in the agile development literature; indeed, in searching the literature, we found numerous studies that mention the term. However, these studies often assume what collaboration means, do not provide clear and consistent definitions, and rarely elaborate on the dimensions that constitute collaboration. For instance, Highsmith (2012) defines collaboration as the ability to work jointly to produce results, share knowledge, or make decisions (Highsmith, 2002). Customer collaboration means that all players—the sponsor, customer, user, and developer—are on the same team (Highsmith & Cockburn, 2001), although this assertion may be questionable for large projects. Other scholars define collaboration as constant and intensive interaction (Hildenbrand, Rothlauf, Geisser, Heinzl, & Kude, 2008). Cummings, Espinosa, and Pickering (2009) relate collaboration with coordination and argue that coordination tools are more effective than direct communication with more predictable and routine tasks.

Historically, agile methods were first employed in small, co-located projects (Bose, 2008; Dyba & Dingsøyr, 2008), and these methods emphasized a co-located customer representative (Larman, 2005). The agile projects managed customer collaboration relatively easily because the small team size (Beck, 2000) facilitated effective communication among developers and between the developers and the customer representative who was available for requirements clarifications. One could also easily assess customer collaboration by using a small number of indicators and without precisely conceptualizing or rigorously understanding customer collaboration’s underlying dimensions. For example, Chow and Cao (2008) measured customer involvement (three items) as “good customer relationship”, “strong customer commitment”, and “customer having full authority” (p. 963). Misra, Kumar, and Kumar (2009) measured customer collaboration as “in our projects, customers closely collaborate with the development team members” (p. 1886) and customer commitment as “in our software development projects, customers are committed to the project, i.e., they are motivated, active, and consider themselves to be responsible elements of the project” (p. 1887).

In the introduction to a special issue on flexible and distributed systems in Information Systems Research, Agerfalk, Fitzgerald, and Slaughter (2009) question the meaning of “working together”. They state (p. 326): “Collaboration needs practical, hands-on study that will distinguish it from the others and consideration of how it’s done when done well. We need language and analysis that will change it from the buzz word du jour to a usefully technical term of art.”. Agile software development practitioners often view collaboration as a subjectively defined capability rather than an empirically quantifiable process capability (Fontana, Fontana, da Rosa Garbuio, Reinehr, & Malucelli, 2014). The main reason for this subjective view among practitioners and even among academic researchers is the lack of systematic understanding of the nature and underlying dimensions of collaboration in agile software development. The absence of a well-defined conceptualization for and underlying dimensions of collaboration hinders researchers’ ability to develop a formal measurement of collaboration and limits practitioners’ ability to create effective mechanisms to manage the specific facets of collaboration to improve agile software development.

We especially need to operationalize and measure collaboration for both theory building and practice related to agility. The contemporary software development environment has changed significantly since the Agile Manifesto was proposed. In the ninth annual state of the agile survey (Versionone, 2015), 94 percent of the respondents indicated that their organizations used some agile practices. Agile adoption is mature, and large projects now also use agile methods (Larman & Vodde, 2010; Leffingwell, 2007; Turner et al., 2013). A scaled version of the popular scrum method (see https://www.scrum.org/Resources/The-Nexus-Guide) that has been deployed successfully (Heikkilä et al., 2015) also exists. However, large projects are complex (Batra, Xia, VanderMeer, & Dutta, 2010; Xia & Lee, 2003) and present a context that
makes collaboration between the customer and the development team much more difficult and yet much more critical than with collocated teams. Larger projects involve a more complex set of interdependencies among many facets of the software development process, and traditional agile practices derived from the original Agile Manifesto and principles cannot address those complex interdependencies adequately. In order to achieve agility, one needs to manage the increased levels of complexities and interdependencies through more effective collaboration.

In addition to increased complexity that results from a larger project size, software projects need to address rapid changes that frequently occur while creating value for customers (Grover & Kohli, 2012; Prahalad & Ramaswanny, 2004). The business customer and software development team need to work together to continuously identify and implement new solutions as the environment changes (Babb & Keith, 2011; Zwass, 2010). The need for both sides to work together on a continual basis implies that not only the development team but also the customer is responsible for initiating and facilitating collaboration (Jaakkola & Alexander, 2014). Customer collaboration in traditional software development projects is typically driven by the developer, and the customer role is quite limited (McLeod & MacDonell, 2011). In contemporary software development, the focus needs to shift from the often one-sided customer collaboration to a new two-sided collaboration to accommodate the increasing levels of complexity and dynamics present.

Outsourcing also drives project complexity due to the physical and temporal distance it creates between the various parties (Batra, 2009; Bose, 2008). Collaboration—especially in outsourcing—can cause issues when the client and the vendor have different organizational or national cultures (Swar, Moon, Oh, & Rhee, 2012). These days, better communication tools somewhat address the physical distance (Bose, 2008). Further, sometimes, the physical and temporal distance may not be a critical factor, such as in Japanese-Chinese client-vendor arrangements because of the long-term working relationships and overlapping time zones. Nevertheless, contextual and cultural drivers affect the nature and strength of outsourcing collaborations.

Overall, we find a theoretical gap: on the one hand, we lack agreed-on conceptualizations of collaboration that we can draw on to extend our customary agile methods to deal with the emergent complex interdependencies in the current large distributed agile software development projects. On the other hand, while both practitioners and researchers have recognized that collaboration is a critical challenge in scaling up agile methods to the large distributed software development context, little research provides clarity on conceptualizing collaboration and delineating its underlying dimensions. We bridge this gap by using the evolved grounded theory-building approach to reveal the core aspects and dimensions of collaboration in outsourced software development projects. By identifying the key aspects and dimensions that emerged from the analyses, we conceptualize collaboration for the scaled-up agile methods. Specifically, we address two research questions:

**RQ1:** What is the nature of collaboration in outsourced agile projects?

**RQ2:** What are the dimensions and subdimensions of collaboration in agile projects?

By answering these questions, we contribute to the agility literature by strengthening the link between collaboration and agility. As such, agility represents a major component of our study.

We studied five companies that had incorporated at least some degree of agile software development. Prior to the selection, the second author had conducted separate field studies using interviews and focus group approaches in Chinese outsourcing companies to study how the companies evolved over time, which included their software development capabilities. He selected these five companies based on a set of criteria to ensure that the sample represented a spectrum of agile adoption, client diversity, vendor size, and project size. We conducted two rounds of interviews with the same participants to progressively gain an understanding about the nature and the dimensions of collaboration. In the first round of open-ended interviews, we employed a generic questionnaire on agile software development. From analyzing the data, we found that the collaboration construct appeared as a core category of agile development capabilities. In the second round of interviews, we focused on targeted questions that led to dimensions and subdimensions of collaboration for agile development.

Our results indicate that collaboration is a broader and deeper construct than what the agile development literature has reported. According to our results, collaboration has five key dimensions: mutual benefits, engagement, coordination, knowledge sharing, and communication. Each dimension captures a different aspect of collaboration that is critical to the success of agile development. These dimensions have
subdimensions and are formative rather than reflective of collaboration. In general, a high score on each dimension will result in better customer collaboration. For example, more agile companies will not only establish a high-level engagement but also back it up with coordination mechanisms such as progress reporting.

2 Data Collection, Interview Protocol, and Company Profiles

We broadly focus on examining how Chinese vendors, who consider collaboration as a key guiding principle, appropriate agile practices. In the past, the Chinese software vendors exclusively worked with Japanese clients and had a strong collaborative relationship; however, in recent years, they have also started working with the United States and European clients. To understand the constructs that affect agile software development, we developed and employed an interview protocol with semi-structured questions to guide our field case studies. The interview protocol included questions on the agile values and principles as listed in the Agile Manifesto and questions on generic factors reported in the software development surveys (McLeod & MacDonell, 2011).

We asked the study participants to indicate their preferences for the Agile Manifesto values and principles so that we could gauge the extent to which they adopted agile practices. We included project characteristics such as purpose, size, scope, budget, and schedule to understand the contextual factors. Based on Lee and Xia (2010), we included questions on the degree of requirement changes so that we could verify the agility aspect of the projects. We included questions about time-boxed development, iteration length, and formal learning workshops to better understand how the vendors used agile methods. We also included several other contextual and general questions, such as questions about what agile values and principles the vendors adhered to, the agile methods they adopted, the length of time that they had used the methods, general principles that they believed to facilitate agility, and key indicators of agility. Other questions covered in the interviews included: “Was the project marred with ambiguous goals, poorly defined requirements, resource shortages, skill limitations, or technical limitations?”. We also asked questions about many other issues such as incorporating agile principles in large projects, late changes, top management support, user involvement, budget, schedule changes, functionality, project management, motivating developers, conflict, and culture.

We conducted two rounds of interviews with managers from five projects of varying sizes from five medium- and large-sized software companies that had offices in various cities in China. We audio-recorded and transcribed each interview. To ensure anonymity, we assigned pseudonyms (R1 through R5) to the study participants. In the quotes, R1 represents company C1, R2 represents company C2, and so on.

The main criteria that we used to select the study sites included the development methodologies the project used, size of the company, type of clientele, and contract types. In terms of software development methodologies used, companies ranged from least agile (C1) and most agile (C5). In terms of size, the companies varied in size from small (e.g., C3) and medium (e.g., C2 and C5) to large (C1 and C4). In terms of type of clientele, companies ranged from Japanese and China (C1 through C4) to US/Europe/North America (C2, C4, and C5). In terms of the type of contract, companies ranged from stable long-term contracts (C1, C3) and mixed (C2, C4) to flexible and market-based contracts (C5).

Company C1 had over 1,000 employees, focused on software for financial services, and had a long-term relationship with Japanese clients. Company C2 employed a co-sourcing model and offered customized software development services in finance, manufacturing, and retailing. In addition to Japanese clients, the company had some clients in the USA, Canada, Europe, and Australia. Company C3 was relatively small and involved in developing software such as online education products, supply chain logistics management systems, and automotive industry software. It provided outsourcing services to domestic, Japanese, and other overseas clients. Company C4 had over 5,000 employees, and provided services to domestic, Japanese, and other overseas clients. Company C5 was medium-sized and did not target Japanese clients but focused more on building strong, collaborative relationships and providing high-quality software. The mission and vision statements of company C5 clearly indicated a strong and explicit commitment to agile development methods.
3 Methodology

We did not start with predefined theories; rather, we employed the evolved grounded theory approach that Corbin and Strauss (2014) stipulate to analyze the qualitative data. One may interpret theory development as denoting a set of well-developed categories (themes, concepts) that are systematically developed and interpreted regarding their properties and dimensions and then related to each other (Hage, 1972). At times, researchers have used the grounded theory approach to generate concepts that one can employ when building theory (Urquhart, Lehmann, & Myers, 2010). Although Matavire and Brown (2013) discuss different ways of employing the grounded theory approach, all grounded theory research projects share the idea that one explores a general area of interest without specific research questions or hypotheses. Grounded theory employs the constant comparison approach, which requires one to break down data into manageable pieces and compare each piece for similarities and differences (Corbin & Strauss, 2014). One groups similar data together under the same conceptual heading. The constant comparison requires one to continually compare indicators to refine the emerging concepts as one generates concepts (Adolph, Hall, & Kruchten, 2011). Through further analysis, one groups concepts together to form categories, which are sometimes referred to as themes (Corbin & Strauss, 2014). Eventually, one integrates the different categories into a core category. Table 1 below illustrates why we conducted our two interview rounds and the coding approaches we used to analyze the resulting qualitative data.

<table>
<thead>
<tr>
<th>Purpose</th>
<th>Open coding</th>
<th>Axial coding</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>First round of interviews</strong></td>
<td><strong>Identify key principles adopted in practice</strong></td>
<td>Analyze first-round interview transcripts to assign emerging concept names to indicators of agile principles adopted in practice</td>
</tr>
<tr>
<td><strong>Second round of interviews</strong></td>
<td><strong>Define key dimensions of collaboration and their subdimensions</strong></td>
<td>Analyze second-round interview transcripts to assign emerging concept names to indicators of dimensions and subdimensions of collaboration</td>
</tr>
</tbody>
</table>

In the first round of interviews, we prepared a protocol with semi-structured and open-ended questions based on issues deemed important in agile software development. We conducted the first set of interviews in five companies that we selected primarily based on their adherence to agile practices, size, type of clientele, and contract type. We audio-recorded all interviews with the participants’ permission and transcribed them. We coded the interview transcripts using NVivo. Following the coding approach that Corbin and Strauss (2014) suggest, we used two phases to code the interview data: open coding and axial coding. In the open coding phase, we assigned the indicators temporary concept names. As the coding continued and we examined the indicators, we compared concept names for similarity. An indicator could be a sentence, paragraph, or even more than one paragraph that represented one coding unit. In the axial coding phase, we grouped similarly named concepts together and gave them category names. We further grouped similar categories into higher-level categories. Three prominent high-level categories emerged: cooperative framework, coordination, and interactions. We used a core category—collaboration—to represent the three prominent high-level categories. Minor categories included context, culture, and outcomes, but we did not consider them because, according to the evolved grounded theory recommendation, once the core category is defined, it should be the primary focus for analysis.

Given that we focused on identifying the key agility principles adopted in practice in the first round of interviews, the questions did not specifically relate to the different dimensions of collaboration. From analyzing the first-round interviews, we conducted a second set of open-ended interviews to ensure that we asked detailed questions about the underlying dimensions of collaboration and reached theoretical saturation. We analyzed the second round of interviews based on open coding and axial coding using NVivo. A more differentiated set of collaboration dimensions emerged: mutual benefits, engagement, coordination, communication, and knowledge sharing.
4 Results

As we state in Section 3, five categories of collaboration emerged from our analyzing the data: mutual benefits, engagement, coordination, communication, and knowledge sharing. The agile development literature has primarily focused on the engagement dimension, but our analysis revealed more dimensions. For example, we found that collaboration needs to provide mutual benefits: the vendor depends on the client, but the client also depends on the vendor. This finding explains why we use the term “collaboration” instead of the unidimensional “customer collaboration”.

We found that task-related coordination is an important element of collaboration. If engagement denotes strategic collaboration, coordination denotes task-based collaboration. Engagement and coordination seem to represent the essence of collaboration in an agile environment; however, based on analyzing our data, we suggest that mutual benefits, knowledge sharing, and communication are additional aspects of collaboration. Mutual benefits provide the reason for establishing a collaborative arrangement. Knowledge sharing indicates the strength of the collaborative relationship and reveals the extent of trust between the two parties. Communication deals with conveying information and understanding the views and opinions of others.

4.1 Mutual Benefits

For collaboration to work out, all parties must gain some benefits and, in return, must provide some value to the others. For example, the client may expect low costs and access to a capable vendor, while the vendor may expect a profit. A contract usually formalizes the agreement. However, the contractual terms are moderated by the level of trust and past experience between the client and the vendor. With a new client-vendor relationship, the two parties tend to have a low level of trust in each other, and the contract between them would need to be more detailed. With an established client-vendor relationship, the contract tends to have more tacit, trust-based elements. Furthermore, with mutual benefits, both parties share a project’s risks; thus, the vendor may absorb a certain amount of costs that stem from requirement changes, but the client contributes to costs resulting from major changes. The client and the vendor share the mutual goal of earning value and limiting risks from the collaboration.

4.1.1 Contractual Framework

Contractual agreements define each party’s benefits and responsibilities. For a first-time project, the contractual terms tend to be formal and detailed. As one participant said:

Rule-based terms will be removed in a long-term project, but for a new project, the terms must be very specific. (R4)

As the parties gain trust, they can reduce formal and detailed contractual terms. For example, because of the nature of Japanese-Chinese relationships (more common in our sample), the two parties did not always formally detail contract terms. Once the parties established a partnership, they built contracts more on trust than formal and detailed terms. As one participant said:

After the customer had become our partner, the supervisory role of the contract was reduced. Now, certain detailed terms will not be specifically documented. (R2)

4.1.2 Risk Sharing

In an agile software development environment, requirement changes are the norm. The changes affect project cost and schedule. With major changes, the contract will usually specify who pays for them. However, in the arrangements between Japanese customers and Chinese software development vendors, the long-term nature of the contracts means that it is not always clear who is responsible, and the parties address issues as the situation demands. To establish trust, the vendor may have an arrangement with the client that the vendor bears minor costs with a cap on the amount. The Chinese vendors felt that the profit margin was relatively low, and they needed contractual mechanisms to control their risks. As two participants said:

The company will handle the change for free if its budget is below three man-months; otherwise, it will be written into the change management plan and will be discussed with the client for a mutually beneficial plan. (R1)
If the project size increases, how to calculate the increased cost will be clearly defined in the contract. One may think this is risk sharing, and we think this is what the customer should do. The profit from Japanese outsourcing is low. The client should pay for the extra cost. (R2)

### 4.2 Engagement

If mutual benefits provide formal reasons for working together, engagement provides the relational aspect of collaboration. Engagement depicts a firm’s degree of involvement and commitment. Involvement indicates that the client and the vendor will participate jointly in tasks and will both contribute to project’s completion. Commitment indicates that the client and the vendor will continue to collaborate despite hindrances or limited success.

#### 4.2.1 Involvement

Involvement over time can build trust and commitment. Vendors should encourage customer involvement so that the vendor can address any requirement changes and problems can promptly. In our interviews, when the practitioners discussed collaboration, involvement was usually the first factor they discussed. As several participants said:

*The development team will consult with the customer before making any developmental related critical decisions. The team will ensure that the final outcome meets their mutual interests.* (R2)

*Allow the customer to participate more. The more opinions the customer contributes, the better the customer will know whether our product development meets their expectations, and it will be easier for the customer to adjust the development plan.* (R5)

*The better way is to chat on the Internet every day with your client so that you can resolve the problems as soon as you come across them. Maybe the best (approach) is to have one client especially someone who understands the customers’ demands to work with us, but it’s unrealistic in the outsourcing area.* (R5)

#### 4.2.2 Commitment

We found that vendors commonly seemed to aim for a follow-up project. A client satisfied with a project’s outcomes will more likely trust the vendor for future projects. With each successful project, the client may be more willing to commit to the vendor. Each vendor seemed to be following this escalating-commitment strategy for improving engagement with the client and promoting it into a partnership, which indicates the highest level of collaboration. As several participants said:

*At the beginning of the project, the client signed a contract for a subsystem as a tentative collaboration, so if we did well in it, we could get the contract to the follow-up projects. As for the clients, they also needed a stable partner so that they could pay more attention to their business.* (R1)

*We started to collaborate with this client from 2010. The partnership was formally established in 2012.* (R2)

*If it is a one-time collaboration, we care only about the product. For long-term collaboration, we focus more on long-term objectives and may not care only about the immediate interests. We also need to consider the compatibility of the follow-up project.* (R3)

*Once we have an intention to collaborate with the client, we start with some small projects. Then if the customer has new requirements, they will ask us whether we have the ability to deliver the new requirements.* (R4)

### 4.3 Coordination

Information systems are complex endeavors and require one to manage links among the various tasks and subsystems. Coordination concerns managing tasks, people’s relationships, and dependencies and assessing the progress of different units. Coordination becomes especially important as the project scales up. In agile development, the conclusion of iterations of multiple teams and the consequent integration can serve as a mechanism to enforce the assessment and management of dependencies and backlogs.
As one participant said: “Coordination between the customers, the team, and the senior managers is critical—50% of my time is spent on coordinating the relationships among these three parts” (R4).

4.3.1 Tool-based Dependency Management
Coordination software can help a vendor manage dependencies and workflows. These software tools allow a project manager to ensure that different parties can identify blockages and address impediments. The tools allow the customer to perform project management tasks such as examining who is working on what task. As one participant said:

*We will typically use a whiteboard if it is an agile development project. However, because we are doing offshore development, we will use some remote management software to coordinate with the customers. We are using Jira, an agile task management system and tracking system to coordinate with the customer. This system is very comprehensive and convenient. Jira can do the source code management, publishing, bug tracking, task management, workflow management, and all the functions required for development. (R4)*

4.3.2 Progress Reporting
Progress reports can serve as monitoring and control mechanisms that can facilitate coordination. Software tools can enable planning, progress tracking, and schedule control. We found that Japanese clients liked to stay abreast of progress so that they could assess progress on activities and monitor key indicators. Thus, they took the initiative to coordinate at the task level. For example, if the development velocity becomes unstable, the client can initiate a meeting to determine the issue. The customer may have an onsite member or team to monitor the progress and facilitate coordination. As participants said:

*We use quantitative development progress analysis tool to generate the daily status report. Our Japanese client requires daily progress report. We will send the audit indicators such as lines of code, checklist, and code review status. (R2)*

*Broadly speaking there are three major practices. The first is regular reporting. Depending on the project characteristic, the reports may be monthly, weekly, or daily. The second practice addresses progress control, schedule control, and progress reporting. This performance assessment is needed to report to the customer. It may be in written or oral format. Third, the customer will create a development plan and share it with the end user and with us. (R3)*

*The customer, as the project owner, is very concerned about the progress of the project. They are monitoring the progress through the project management system. Also, the system will automatically generate reports and send them to the customer. For example, if the team's development velocity is unstable in three days and surpasses the indicators set by the customer, the system will automatically generate a report to the customer. (R4)*

4.3.3 Iteration
Iteration provides a checkpoint for a vendor during development that can be used to reconcile the project’s status and manage dependencies among the project’s stakeholders; thus, it represents an important aspect of coordination. Contextual factors can affect how long iteration occurs for. Company C1 iterated for more than a month, but the company left the exact duration undefined; in this case, the development was more structured than agile. Companies C2 and C3 had one-month long iterations, whereas C4 and C5 had two-week long iterations. As agility level increased or project size decreased, the iteration duration decreased. As one participant said:

*We hope the iteration to be two weeks, but we have to coordinate with the client. The longest duration will be less than one month and will be based on our coordination with the client, the complexity of the project, and the clarity of their demands. The less clear the demands are, the shorter the iteration would be, and it will take the customer shorter time to give us their feedback. (R5)*

4.4 Knowledge Sharing between the Client and Vendor
Collaboration requires that the parties leverage each other’s strengths (typically, their expertise and procedural know-how). The client may have the business knowledge to share with the vendor, and the vendor may have the technical knowledge to share with the client. Team members from each party may
also share knowledge among themselves. As the client and the vendor trust each other more, they may sometimes proactively share knowledge to achieve goals beyond a single project.

Vendor and client firms need to share operational knowledge for projects to succeed; however, managers may visit each other’s premises to gain knowledge about each other’s domain, techniques, and culture. Some of this learning is tacit as one party gains a deeper understanding of the workings of the other through interactions and observations. As the following quotes indicate, the level of knowledge sharing increased as the level of agile development commitment increased from companies C1 (the least agile) to C5 (the most agile).

We knew little about the implementation method of the project, so before we built the team, one of our senior managers stayed in Japan for nine months to learn about relevant techniques. It took us another half-year to improve the team’s techniques and their understanding of the client’s business. (R1)

In the early stages of the project, the customer provides some vision and functional explanations, including a description of some specific functions. With the focus on the business value, the customer explains some operational functions and processes to the development team. The customer provides support during the development process. This is essentially an offshore outsourcing project, so it is the customer’s responsibility to provide the development team needed explanations. (R4)

We accumulated a lot of experience through work with consulting firms. As an example, we did not know how to do a fixed-price project at the beginning. By collaboration with the consulting company, we learned how to deal with the fixed-price project from them; such as how to define a clear scope at the initial stage of the project. We will use this knowledge to manage the fixed-price project. (R5)

4.5 Communication

Given the iterative and interactive nature of the agile development process, there are ample opportunities for information exchange and the consequent understanding and reconciliation of diverse perspectives of the team members. One may describe communication as the frequency and diversity with which people exchange information to gain understanding and achieve consensus. Because collaboration needs information exchange and mutual understanding, collaboration involves several communication elements.

4.5.1 Liaison

One way to establish a communication bridge is to establish a liaison role, whether internal or external. A liaison mechanism can improve communication’s effectiveness by improving how accurately one translates information, especially during the initial stages of a project. The vendor’s liaison group can have a face-to-face conversation with the client and then pass on the information in a language more suitable to the offsite vendor development team. As one participant said:

A team of 1-7 employees will stay at the customer’s site. There is also an offshore team responsible for development. Only the onsite team can communicate with the offshore team. The onsite person or team functions as a bridge between the two sides. For this project, sometimes we needed an onsite team initially, but we did not need them later. The onsite staff will return to the offshore site after the development attains relative stability and the customer will then directly communicate with the offshore team. (R4)

4.5.2 Information Exchange

Information exchange is what we normally mean when discussing communication. In today’s agile development environment, it can manifest in many different ways. What mode one uses to communicate depends on factors such as interaction richness and cost efficiency. Outsourcing, while common today in software development, causes communication issues due to the physical and temporal distance it creates between parties. While an in-person face-to-face method works best for exchanging information, outsourcing often does not allow it to occur; thus, other (possibly less effective but more efficient) communication modes that take advantage of the communication technologies have arisen to solve the issue. The face-to-face preference listed in the Agile Manifesto is more relevant for achieving shared understanding but not for daily communication. As one participant said:
We communicate with client's development team daily. It is a collaborative development project so that we will communicate every day. If we need to express our idea more clearly, we will initiate a telephone or video conference. It depends on the content of the communication. Video conference is the best way to share our opinion exchange ideas. If we can present the working demo during the video conference, the effect will be better. However, sometimes if we need to deliver documents or other official files to the client, email may be a better way to communicate. Email communication offers us some buffer time to think before responding. Video conferencing is more about timely feedback. (R5)

4.5.3 Shared Understanding

Clients and vendors have different competencies. In general, clients have the domain expertise, and vendors have the technical expertise. This mismatch can result in misunderstandings. The way to eliminate misunderstandings and uncertainties about the software product requires that the team members of the two sides communicate with each other to reach a shared understanding. Whereas information exchange focuses on the exchange medium, shared understanding focuses on attaining consensus. In general, vendors and clients need face-to-face communication conducted in a meeting-like mode to attain shared understanding quickly and effectively. As two participants said:

*Early communication is very important. There are many uncertainties in the early stage of the project, and we must verify with the customers. A couple of months later, after the customer and we have developed a common understanding of the development style and process, it is much easier to understand and work on the development of functionalities.* (R4)

*We prefer face-to-face communication because it is more efficient. It is about timeliness. If they want to have a discussion with other members, they can do it freely without any delay. We also have a fixed communication mechanism, such as daily meeting each morning. Also, we have code review meetings. There is a review team meeting. Our communication is very frequent compared to a non-agile team. We do not use formal documents. For some of the new requirement from customers, we will present the working software to customers. The customer prefers to finish the product as quickly as possible. So timely communication is critical for us because we do not have many writing documents to support the development.* (R5)

5 Discussion

In this study, we address two research questions: 1) what is the nature of collaboration in outsourced agile projects and 2) what are the major dimensions and subdimensions of collaboration in agile projects. While other researchers have studied collaboration in the agile context to also answer the first question, we provide a field study-derived conceptualization to extend their definitions. The agile development literature has not adequately addressed the second question. Researchers have sometimes referred to similar dimensions (e.g., collaboration as involvement), but they have rarely proposed any subdimensions. With that said, studies on agile projects in the literature have usually focused on small projects (Dyba & Dingsøyr, 2008), and, thus, the researchers had no acute need to suggest detailed measures of collaboration for co-located, small projects. However, contemporary agile projects now come in all sizes. We need to integrate the extant literature on collaboration in agile development and other areas and the proposed dimensions and subdimensions of collaboration from this study. Table 2 provides a link between our findings with those from some empirical studies from the agile development literature (especially where the two findings are consistent). However, one can also see that the literature has a piecemeal view of collaboration, whereas we provide a systematic view of collaboration.

Researchers have defined collaboration in various ways. Highsmith and Orr (2011) define it as an act of creating together based on trust and respect. Vlietland and van Vliet (2015) define it as the process of two or more people’s working on a task, and they distinguish collaboration from both communication and coordination. Based on analyzing our data, we define collaboration more broadly than either of these authors because we extend the actors from people to parties and from tasks to projects. Furthermore, we found that, in large, distributed, and agile projects, one needs to consider both communication and coordination as aspects of collaboration. As we mention in Section 1, agile has permeated organizations and projects of all sizes and all locations. Agility requires collaboration. Based on our results, we define collaboration among parties as “a set of engagement activities through communication and coordination
Communications of the Association for Information Systems 439

for mutual benefits actualized by sharing knowledge and information to accomplish agile software development tasks”.

Table 2. Linking Dimensions and Subdimensions with Key Literature

<table>
<thead>
<tr>
<th>Dimension</th>
<th>Subdimension</th>
<th>Study findings</th>
<th>Key findings from literature</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mutual benefits</td>
<td>Contractual framework</td>
<td>As the parties gain trust, they can reduce detailed contractual agreements. Our findings indicate that a contract ensures mutual benefits and fosters collaboration.</td>
<td>The Agile Manifesto stipulates that customer collaboration is more important than contract negotiations. Bose (2008) found customer collaboration over contract negotiation in four out of 12 cases he studied, which supports the need for contracts.</td>
</tr>
<tr>
<td></td>
<td>Risk sharing</td>
<td>The vendor expects the client to share the risks.</td>
<td>Pilios (2015) has proposed agile contracts that can deal with risk sharing.</td>
</tr>
<tr>
<td>Engagement</td>
<td>Involvement</td>
<td>Involvement is a vital aspect of collaboration.</td>
<td>Research has identified involvement as an important aspect of collaboration in (Hoda, Noble, &amp; Marshall, 2011).</td>
</tr>
<tr>
<td>Commitment</td>
<td>Commitment</td>
<td>Commitment is a much stronger indicator than involvement of collaboration. Vendors aim to escalate clients’ commitment for better engagement and relationship.</td>
<td>Chow and Cao (2008) employed commitment as one item to measure collaboration. However, Misra et al. (2009) measured commitment separately from collaboration.</td>
</tr>
<tr>
<td>Coordination</td>
<td>Dependency management</td>
<td>Coordination software can facilitate management of dependencies and workflows, and enhance task-based collaboration.</td>
<td>Sharp and Robinson (2008) define coordination as management of dependencies and assert that collaboration mechanisms focus more on progress reporting and not enough on managing dependencies.</td>
</tr>
<tr>
<td></td>
<td>Progress reporting</td>
<td>Progress reports can serve as monitoring and control mechanisms that can facilitate coordination.</td>
<td>Vlietland and van Vliet (2015) found that a lack of information visibility in the chain results in coordination problems. Progress reports can improve this visibility.</td>
</tr>
<tr>
<td></td>
<td>Iteration</td>
<td>Iteration provides a checkpoint during development that one can use to reconcile the status and dependencies.</td>
<td>The agile development literature has found extensive support for iteration. However, it does not directly review iteration as an element of coordination or collaboration.</td>
</tr>
<tr>
<td>Knowledge sharing</td>
<td>Sharing know-how between the client and vendor</td>
<td>Knowledge sharing pertains to exchange of operational knowhow, domain, techniques, and cultural values through interactions and observations.</td>
<td>Ghobadi (2015) conducted a literature review on knowledge sharing and found people-related, team perception, task-related, and technology-related drivers.</td>
</tr>
<tr>
<td>Communication</td>
<td>Liaison</td>
<td>A liaison mechanism can improve the effectiveness of the communication by improving how accurately one translates information.</td>
<td>The role of the product owner is partially to liaise with external project stakeholders (Matook &amp; Maruping, 2014). Many studies discuss the role of a product owner; however, the notion of the liaison representative is mainly for translation and information transfer.</td>
</tr>
<tr>
<td></td>
<td>Information exchange</td>
<td>Communication technologies complement face-to-face interaction.</td>
<td>Several studies such as Mishra, Mishra, and Ostrovská (2012) and Molokken-Ostvold and Furulund (2007) emphasize information exchange.</td>
</tr>
<tr>
<td></td>
<td>Shared understanding</td>
<td>Shared understanding is essential for eliminating misunderstandings and uncertainties about the software product.</td>
<td>Shared understanding can result in shared mental models, which are important for achieving goals (Yu &amp; Petter, 2014).</td>
</tr>
</tbody>
</table>

We identified five constructs as key dimensions of collaboration. We posit that one should view collaboration as a formative rather than a reflective construct. The agile literature has sometimes equated collaboration with relationship quality or customer involvement, but the construct seems to have more
dimensions. Engagement may be the equivalent of relationship quality, while coordination may depict an estimate of customer participation at a regular, task-based level. The other three dimensions—mutual benefit, knowledge sharing, and communication—are all important in capturing critical aspects of collaboration in agile development. For example, mutual benefits are an intrinsic component of collaboration. Customer collaboration is vital, but, if the business value has to be co-created (Kohli & Grover, 2008; Prahalad & Ramaswamy, 2004), then we need also to consider the reciprocal notion “vendor collaboration”. We found that a contractual framework does not compete with but can improve collaboration because of clarity regarding budget and schedule adjustments. Agile software development puts a lot of onus on the vendor, especially in dealing with requirement changes. A contract can limit both client and vendor liabilities and define a framework for sharing risks. Collaboration can only last if the mutual benefits are reasonably explicit (even if some are tacit). Knowledge sharing and communication can reveal the quality and depth of collaboration.

Coordination emerged as a term that we assigned to a collection of task-level concepts that we coded as part of the collaboration. The agile development literature usually distinguishes between collaboration and coordination and sometimes ignores coordination when discussing collaboration. For example, Misra et al. (2009) define collaboration as working together to accomplish a task and discussing with other people solutions to difficult problems, but they define coordination as the harmonious adjustment or interaction of different people or things to achieve a goal or effect. Moe, Aurum, and Dyba (2012) state that agile methodology has changed the nature of coordination and collaboration among various stakeholders, which suggests that the authors jointly consider the two constructs even though they imply a separation between the two. Our data analysis reveals that this separation is not necessary: coordination is a vital aspect of collaboration because it pertains to the continual task-based level collaborative activities that are vital to agile development.

Kotlarsky and Oshri (2005) define collaboration as a complex and multidimensional process described by constructs such as coordination, communication, relationship, and trust. In Sections 5.1 to 5.5, we compare our findings of the key collaboration dimensions with those from the literature. We order the dimensions based on their subjective relevance as revealed by data analysis.

5.1 Engagement

We define engagement as the degree of each involved party’s involvement and commitment. Engagement depicts a significant level of active, nontransactional involvement between the client and the vendor (Jaakkola & Alexander, 2014). The concept of customer engagement is a recent attempt to aggregate multiple ways of customer behaviors beyond transactions (Brodie, Hollebeek, Juric, & Ilic, 2011; Verhoef, Reinartz, & Krafft, 2010). There is considerable interest in the potential to engage customers and customer communities in "coproduction" or "co-creation" to enhance business performance or customer value (Prahalad & Ramaswamy, 2004).

Researchers have identified the lack of customer involvement as one of the biggest challenges that agile teams face (Hoda et al., 2011). Inadequate customer involvement in these agile projects leads to adverse consequences for the self-organizing agile teams, including problems in gathering, clarifying, and prioritizing requirements and problems in securing feedback. One literature review revealed that customer involvement is a software development success factor (Inayat, Salim, Marczar, Daneva, & Shamshirband, 2014). An onsite customer representative enhances involvement, which results in improved project success (Matook & Maruping, 2014). Lack of customer involvement can affect customer communication, which, in turn, can affect how the vendor understands requirements (Alzoubi & Gill, 2014). Swar et al. (2012) found that relationship quality, which was a key determinant of IS/IT outsourcing success, comprised cooperation, trust, and mutual understanding.

While involvement is the first aspect, commitment is the second aspect of engagement. Projects are temporary work systems that focus on producing a set of products/services after which they cease to exist (Alter, 2003). Escalating commitment between the client and the vendor is a means to improve collaboration until the relationship can mature into a partnership so that it reduces transactional and monitoring costs and attains business stability. Japanese clients specifically prefer long-term, trusted relationships, but reaching a stable relationship is a process of escalating commitment that the vendor facilitates through providing a high-quality service.
5.2 Coordination

In the context of agile development, coordination can be defined as the synchronization of tasks and people through the management of dependencies, the evaluation of task progress, and the periodic reconciliation of the artifact in development. Coordination indicates the spatial and temporal synchronization of overt behaviors of two or more people so that those actions fit together into an intended spatial and temporal pattern (Arrow, McGrath, & Berdahl, 2000). Van de Ven et al. (1976) identify task uncertainty, task interdependence, and the size of the work unit as fundamental determinants of the coordination mode. In agile development, task uncertainty is a common issue given that one does not know the software artifact the beginning of the project. The size of the work unit increases as agile development addresses larger projects. Task interdependence is correlated with the project size and the number of parts. Iteration and continuous integration can mitigate these issues. As the size of a project grows, coordination becomes critical and requires sophisticated software tools to manage the progress.

In a software development project, a high level of coordination leads to many benefits such as shorter development cycles, cost savings, and better-integrated products (Espinosa, Slaughter, Kraut, & Herbsleb, 2007). Vlietland and van Vliet (2015) found that coordination is an important issue in collaboration; even some of the other issues they identified, such as mismatches in backlog priority and a lack of information visibility in the chain, pertain directly or indirectly to coordination issues. Sharp and Robinson (2008) found that collaboration mechanisms focus more on progress reporting and not enough on managing dependencies. Our findings concur with their assertion: the respondents focus more on progress reporting rather than on dependency management. The planning and reporting mechanisms of coordination we found may usually correspond to what researchers term an impersonal mode of coordination (Dietrich, Kujala, & Artto, 2013). Mechanisms such as pre-established plans, schedules, forecasts, formalized rules, policies, procedures, and standard information and communication systems exemplify this impersonal model. Coordination software can also allow informal modes of planning and reporting through query language features that can provide filtered information on an ad hoc basis.

Researchers have identified Agile Manifesto principles 1 and 3, which deal with iterative development and customer satisfaction, as the most critical for agile development (Williams, 2012). Iterative development can enhance coordination by providing an artifact that can serve as a basis for feedback and understanding. Sometimes, vendors use the practice of small releases on an ad hoc basis, but the continual component integration ensures that the client can observe and evaluate the artifact as it emerges (Wang, Conboy, & Pikkarainen, 2012).

5.3 Communication

In our study on agile development, we observed three modes of communication: liaison, information exchange, and shared understanding. They emerged as collaboration aspects when we analyzed the data. The literature supports and concurs with our observations about including communication as a key dimension of collaboration.

Communication refers to the imparting or interchanging of thoughts, opinions, or information via speech, writing, or signs (Mishra et al., 2012). It is a dialogue that attempts to balance creativity and constraints through information transfer, transactional process, or strategic control (Eisenberg & Goodall, 2004). Communication is a key factor for agile software development (Beck, 2000; Hummel, Rosenkranz, & Holten, 2013; Mishra & Mishra, 2009). In projects where daily communication between the contractor and the customer facilitates collaboration, the customer experiences a lesser magnitude of effort overruns (Molokken-Ostvold & Furulund, 2007). Communication facilitates appropriate understandings about project scope, project tasks and activities, and project milestones and future goals. The shared understanding can result in shared mental models, which are important for achieving goals (Yu & Petter, 2014). Korkala and Abrahamsson (2007) report that communication is not the key factor; instead, they consider a well-defined customer to be most important for distributed projects. The liaison team can perhaps do both: ease communication transfer and work closer to the customer. In an outsourcing environment, therefore, the liaison aspect of communication may become especially important.

5.4 Knowledge Sharing

Knowledge sharing involves the exchange of expertise between the client and the vendor. One reason why knowledge sharing was more likely to occur between the clients and the vendors in our study may result from the long-term relationship and the ensuing trust and interdependency in the Japanese-Chinese
outsourcing associations. The vendor’s domain knowledge is a key asset for setting up successful client-developer collaboration (Daneva et al., 2013). In reviewing the literature on knowledge sharing, Ghobadi (2015), found people-related drivers such as the knowledge receiver’s mindset and levels of knowledge, transactive memory, and geographical distribution; team perception drivers such as trust and interdependencies; task-related drivers such as project risks and project knowledge; and technology-related drivers such as project methodology and collaborative technologies. These drivers are not components of but are external to knowledge sharing; however, they underscore the role of context, trust, methodology, and technology in fostering knowledge sharing. Thus, knowledge sharing in particular and collaboration in general are affected by other constructs, which should be considered as antecedents in a structural equation modeling study.

5.5 Mutual Benefits

We observed in all our study sites that each party had certain incentives when engaging in a collaborative arrangement. Whether tangible or intangible, mutual benefits reflect a firm’s motivation for maintaining or strengthening a collaborative arrangement. As such, one cannot adequately conceptualize collaboration without considering mutual benefits. When two parties already work as partners, whether formal or informal, the transaction costs are minimal, and the parties have an incentive to co-create business value. This kind of a goal concurs with agile software development, which emphasizes collaboration. Based on the contextual factors, the contractual framework should be tuned to collaboration needs. From reviewing the IT outsourcing literature, Lacity, Khan, and Willcocks (2009) provide an extensive list of motivations for IT outsourcing such as cost reduction, focus on core capabilities, access to expertise/skills, and improving business/process performance. In general, mutual benefits specify the gains and avoidance of losses in a contractual framework or informal agreement, which is moderated by relational factors such as trust and past experiences.

6 Implications and Future Research

6.1 Implications for theory

Although collaboration is an essential focus of agile software development (Moe et al., 2012), we lack systematic measure for assessing collaboration in agile software development. Thus, we need to develop such measures to systematically understand its nature and underlying dimensions. To the best of our knowledge, this study is the first that explores and defines the concept and key dimensions of collaboration in agile development literature. By employing the grounded theory methodology, we provide a framework for conceptualizing collaboration by identifying five dimensions: mutual benefits, engagement, coordination, communication, and knowledge sharing. Each dimension has subdimensions, which one can express as formative or reflective constructs for quantitative studies.

Because agile projects started off as small and co-located (Dyba & Dingsøyr, 2008), the practitioners and the researchers did not initially consider coordination as an important facet of collaboration. Contemporary agile software development involves larger projects, and coordination could be a key role of project leads who visit offshore locations to liaise with the developers and facilitate daily tasks (Bose, 2008). Few agile development research studies have focused on the role of coordination; on the contrary, the management literature has suggested that coordination should be considered as an aspect of collaboration. According to Gulati, Wohlgezogen, and Zhelyazkov (2012), scholars have paid less attention to the critical role of coordination, which focuses on the mechanics of bringing together partners’ contributions, and more attention to cooperation, which is at a more strategic level and deals with benefits and contractual issues. They feel that coordination is as important as cooperation when discussing collaboration.

Similarly, one could examine mutual benefits, communication, and knowledge sharing in more detail to assess if one could measure the subdimensions of these constructs as indicators for measuring collaboration. For example, one could examine the area of agile contracts (Piliös, 2015) to determine how one could measure mutual benefits. One can view specific subdimensions of communication such as dialogue/information exchange, shared understanding, and liaison mechanisms to determine the indicators for measuring collaboration.

By employing an empirical quantitative data-collection and -analysis approach (e.g., structural equation modeling), one could use these aspects to develop a more refined theoretical measurement of collaboration, which prior research has hypothesized to affect agility and other measures of project
outcomes. This study provides a basis for measuring collaboration. One can measure agility by employing Conboy’s (2009) conceptual work. Other project outcomes include measures such as budget, schedule, quality, and team satisfaction. The model could have antecedents such as team autonomy (Lee & Xia, 2010), team competence (Gallagher, Kaiser, Simon, Beath, & Goles, 2010), and contextual factors such as project size, dynamism, and culture (Boehm & Turner, 2004). The resulting model and analysis can provide a better theoretical understanding of agile development.

Our study already provides some evidence that collaboration affects agility. Collaboration patterns differed among the five companies. Although all companies had mutual benefits as a requisite goal, the extent to which they used other collaboration mechanisms depended on their degree of agile development. The least agile company, C1, seemed to have mutual benefits as the primary goal. The most agile companies such as C4 and C5 employed iterations for better coordination, involvement for better engagement, shared understanding for communication, and more knowledge sharing; the intermediate agile companies such as C2 and C3 employed reporting for better coordination, information sharing for communication, and relatively less knowledge sharing.

### 6.2 Implications for Practice

Our results have specific implications for practitioners. They need to view collaboration as a two-way street. Contractual frameworks need to assess relational aspects such as trust and historical dependency. Risk sharing should limit the liability of any one party. The client needs to be involved and committed to the development. Coordination mechanisms can actualize the collaboration efforts at the task level. Parties need to pay special attention to people, tasks, and software component dependencies and progress. If necessary, parties need to establish liaison roles in outsourcing environments. Information sharing and shared understanding facilitate collaboration. Knowledge sharing mechanisms can aid in the co-creation of business value.

Our findings deviate in some ways from the Agile Manifesto principles. For example, the Agile Manifesto does not explicitly delve into coordination or discuss the role of contractual frameworks or mutual benefits. The Manifesto focuses only on face-to-face communication and does not directly address knowledge sharing. However, our findings concur with the Agile Manifesto in terms of engagement’s role as a driver of customer collaboration.

### 6.3 Limitations

This study has two main limitations. First, our research goal shifted from studying agility in general to collaboration in particular, which occurred because collaboration emerged as a core category after our first round of data collection, which employed a generic questionnaire. In the second round of data collection, we asked focused questions on collaboration so that the analysis could reach theoretical saturation. The two-step approach may not follow textbook prescriptions.

Second, because of the study’s inductive nature, we cannot provide a detailed view of the effects of antecedents on collaboration and of collaboration on agility and outcomes. In this sense, one needs to theoretically extend our study. Our study does provide a rigorous typology of collaboration that one can use to build theory. To provide a more complete theoretical understanding, we have begun conducting a separate quantitative study.

### 6.4 Future Research

Future research can use our study to develop measurement scales for collaboration. First, one will need to integrate our dimensions and subdimensions with findings from the agile development literature in particular and the IS and management literatures in general. Second, one will need to test a preliminary scale with practitioner experts. Third, one will need to conduct a pilot test to reveal the validity and reliability issues. After these issues are resolved, the scale will be ready for answering the critical research questions such as what the antecedents of collaboration are and how collaboration affects key dependent variables such as capability, agility, and project success.

Context and culture can be antecedents or moderator variables of collaboration. For example, with an outsourced project, the contractual elements will become more significant. A project’s size can affect coordination mechanisms as the interdependencies become more complex. The duration of relationship between the client and the vendor can affect commitment, contractual elements, and knowledge sharing.
One needs to consider the role of team autonomy, team competence, and trust. One needs to determine the strength of these relationships quantitatively by adopting a structural equation modeling or a similar approach.

7 Conclusion

Customer collaboration is perhaps the most important value of the Agile Manifesto. Some principles that the Agile Manifesto lists refer directly or indirectly to customer collaboration. In this paper, we extend the notion of collaboration by revealing its dimensions and subdimensions in the context of contemporary outsourcing agile software development. Specifically, we extend the notion of collaboration to include both customer and vendor collaboration. We identify mutual benefits, engagement, coordination, knowledge sharing, and communication as five key dimensions underlying the overarching construct of collaboration in agile software development. Our results conceptualize collaboration based on our analyzing data from field study cases. The study provides a first step that can serve as a foundation for further developing meaningful, quantitative measures for effectively assessing collaboration in the context of agile development, particularly in the contemporary organizations that use agile approaches in both small and larger projects.
References


Alter, S. (2003). 18 reasons why IT-reliant work systems should replace “the IT artifact” as the core subject matter of the IS field. *Communications of the Association for Information Systems, 12*, 72-121.


About the Authors


Weidong Xia is on the faculty in the Department of Information Systems and Business Analytics at Florida International University. Prior to joining FIU, he was on the faculty of the Carlson School of Management at the University of Minnesota. His research interests relate to project management complexity and flexibility, healthcare process and analytics, and IT-enabled business innovations. Dr. Xia’s publications have appeared in *MIS Quarterly, Journal of Operations Management, European Journal of Information Systems, Decision Sciences, Journal of MIS, Journal of Information Systems, Communications of the ACM, Communications of the AIS, Journal of Information Technology Management*, and other journals.

Mingyu Zhang is a PhD candidate at the Department of Information Systems and Business Analytics, Florida International University, Miami, USA. His areas of research interest are agile software development, IT outsourcing, and IT project management.