Value Proposition of Agility in Software Development – An Empirical Investigation

Completed Research Paper

VenuGopal Balijepally
Oakland University
balijepa@oakland.edu
Vijayan Sugumaran
Oakland University
sugumara@oakland.edu

Gerald DeHondt
Oakland University
dehondt@oakland.edu
Sridhar Nerur
The University of Texas at Arlington
snerur@uta.edu

Abstract

Agile Development Methods have received much attention since their inception, being regarded as an alternative to plan-based, phased methodologies, which are guided by a waterfall process model. These practices have evolved and developed over time, culminating in 2001 with the Agile Manifesto. Since that time, preferred methodologies, implementations, and best practices have continued to evolve with a focus on doing what works best for the individual company or project. However, the concept of agility in software development has remained quite nebulous, lacking in clarity particularly about its underlying dimensions. In this research we conceive agility in terms of four distinct underlying dimensions. Drawing from the theoretical perspective of holographic organization, we develop a model explaining how each of these underlying dimensions of agility contributes to project value in software teams. We test the model using survey data collected from industry practitioners and discuss findings.

Keywords

Agile Software Development, Business Value, Holographic Organization

Introduction

Ever since its inception, the field of software engineering has struggled to cope with the myriad challenges inherent in software development. Indeed, as early as the 1960s, an engineering approach was conceived to deal with the crisis that had been precipitated by ad hoc programming practices that paid little or no heed to several factors (e.g., analysis, design, customer involvement, collaboration and communication) that are deemed to be critical for the success of software development projects. Early efforts at delivering valuable and usable software on schedule and within budget placed considerable emphasis on upfront, meticulous planning and on a linear sequential approach that posited a stable and predictable world. The principles articulated in the Agile Manifesto (AgileAlliance 2001) and the practices that ensued were largely because of the litany of software project failures as well as the inability of erstwhile methods to deliver high quality software within a stipulated schedule and budget.

Over the last decade, a host of agile methods (e.g., Scrum, eXtreme Programming) have dotted the software development landscape. Consistent with the principles enunciated in the Agile Manifesto (AgileAlliance 2001), these methods place primacy on iterative development, team empowerment and collaboration, adaptive planning, active customer engagement, continuous process improvement, and
value to stakeholders (Cockburn and Highsmith 2001; Highsmith and Cockburn 2001). Further, they eschew unnecessary documentation and rely on self-organizing teams to respond to problematic situations that inevitably emerge during the development process. The belief that developers who are unconstrained by rigid processes will be able to produce ingenious software solutions is implicit in the agile movement. By all accounts, the principles and practices advocated by “agilists” have been adopted or adapted to varying degrees by companies around the world (Dingsøyr et al. 2012).

With the exception of a few empirical papers ([e.g., (Bonner et al. 2010; Grenning 2001; Manhart and Schneider 2004)]) and some anecdotal accounts, the efficacy of Agile Development Methodology (ADM) has hardly been subjected to empirical scrutiny. In particular, there has been little or no effort to substantiate the claims that practices such as iterative development, self-organizing teams, process flexibility (e.g., interchangeable roles), and test-driven development deliver stakeholder value. This is rather surprising, considering the widespread acceptance of ADM. Our study aims to fill this void. Specifically, this research seeks to examine the following main research question: what aspects of agility in software development methods contribute to project value? We use the theoretical perspective of holographic organization (Morgan 1998; Nerur and Balijepally 2007) to explicate the value proposition of agility in software teams and to derive the main hypotheses.

The data for this research were gathered from a survey of industry practitioners, conducted by Ambyssoft, Inc. (Ambler 2013). Respondents were from companies across the globe representing numerous IT roles including developers, Quality Assurance professionals, business analysts and subject matter experts, and project managers. It is believed that this broad group of participants—including managerial and technical professionals, dispersed across multiple organizations of varying sizes, and geographically distributed—will provide a better picture of the overall business value that accrues to organizations by virtue of using Agile Software Development.

The remainder of the paper is organized as follows. We first provide a very brief description of Agile Software Development. This is followed by an overview of the holographic principles of organizational design, which provides the theoretical rationale for the main hypotheses of the study. Subsequently, we discuss the research method, followed by a presentation of the data analysis and the results of hypothesis testing. Finally, we discuss the implications of our research findings and conclude with some directions for future research.

**Agile Software Development**

Agile software development has long been seen as an alternative to the traditional Systems Development Life Cycle or plan-based methodologies. Agile methods, in general, have been well received by those in the systems development community, and have seen increasing use over time (Conboy 2009). Several studies have noted the successful use of agile methods ([e.g., (Bonner et al. 2010; Grenning 2001; Manhart and Schneider 2004)]. Bonner et al. (2010) found that development process agility, a construct that subsumes evolutionary development and process flexibility, was perceived to reduce complexity while being compatible with prior experience, work practices and values. Further, reduced complexity and increased compatibility were found to be positively associated with perceived benefits. Agile methodologies are also purported to imbue flexibility in software development projects, thereby enabling software development teams to perform more effectively (Maruping et al. 2009). Conboy (2009) notes that developers rarely adhere to commercial methods, and that the vast majority use tailored variants or in-house creations. Agile methods are believed to be more responsive to a changing environment, with working software being chosen over extensive documentation. Lee and Xia (2010) emphasize sense-and-respond, cross-functional teams, and continuous adaption. Central to Agile development methods is the idea that many of the development processes cannot be predicted. Agile seeks to address software development in a
Value Proposition of Agility in Software Development

flexible way (Vlaanderen et al. 2011) and should not be viewed as isolated ideas, but as closely interrelated and interdependent (Erickson et al. 2005). Taken together, the advantage of Agile methods lies in providing the ability to adapt and respond to a changing software development environment.

Conboy (2009) has defined agility as 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). Erickson et al. (2005) note that agility is often associated with such related concepts as nimbleness, suppleness, quickness, dexterity, liveliness, or alertness. At its core, agility means to strip away as much of the heaviness, commonly associated with traditional software-development methodologies, as possible to promote quick response to changing environments, changes in user requirements, accelerated project deadlines, and the like. Erickson et al. (2005) reflect that the traditional established methodologies are too set and often too full of inertia to respond quickly enough to a changing environment.

From this, the basis of Agile methods are the implementation of techniques and best practices that work for the individual organization and the project at hand. The resilience of Agile, and its adaptability and willingness to change, to the organization or implementation, has allowed it to evolve and improve over time. Additionally, the nimbleness, quickness, and dexterity mentioned above, allow Agile to be more responsive to customer needs in a fast-paced, ever changing environment. With the weight and inertia of traditional methods, plan-based methods may be slow to change, and reflect state of the art from an earlier time. As Agile methods have developed, best practices have evolved through implementations, and practitioners have changed the preferred techniques, or best practices, of these techniques over time.

This research looks at several areas of Agile Software Development and attempts to delineate the underlying dimensions of agility and their contribution to the project value.

Holographic Principles of Organizational Design

A hologram is a picture constructed with laser based technology where the whole is represented in each of its parts. Thus, if the hologram is accidentally damaged, it can easily be recreated from any of its parts (Morgan and Ramirez 1984). It is believed that the human brain is organized based on holographic principles with memory distributed across different parts of the brain. Extending the hologram metaphor to the organizational context, holographic principles of organizational design suggest ways social systems faced with turbulent environments could be designed for adaptability and responsiveness (Morgan and Ramirez 1984). The five principles underlying holographic design are: building whole into parts, requisite variety, redundancy of functions, minimal critical specification, and learning to learn [also see (Nerur and Balijepally 2007)]. When designing social systems such as groups and organizations for adaptability and high responsiveness required in dynamic environments, the holographic principle of manifesting the whole in each of its parts calls for creating holistic self-organized teams that could learn and adapt quickly to situational demands. The next principle is based on Ashby’s law of requisite variety (Morgan and Ramirez 1984), which stipulates that the internal sophistication and variety of a system should be commensurate with the complexity and variability of its external environment. The principle of redundancy of functions emphasizes the need for social systems to have spare interchangeable resources so that in case of problems in one area, other replaceable resources could be deployed with minimal disruptions to the working of the system. Unlike traditional organizational designs that seek to minimize redundancies and overlap between resources on the grounds of efficiency, holographic design calls for providing some redundancy of functions/resources so that the system has flexibility to respond quickly and effectively to unforeseen demands. The last holographic principle of ‘learning to learn’ is highly critical to the functioning of the system in dynamic environments. It calls for reflection and constant...
questioning of the current assumptions so that the system is able to learn quickly from the environment and from its own actions. Empowering and equipping the system for self-organization is essential for enhancing responsiveness of the system. The principle of minimal critical specification highlights the futility of trying to foresee and plan for all possible demands on an organizational system, particularly in dynamic environments. This cripples innovation and constrains system response. Instead, the system should be specified with minimal essential demands so that it could learn and tailor its response to changing requirements.

Minimum critical specification, which is one of the principles of socio-technical systems (STS) theory, provides the foundation for local autonomy of teams, which in turn, facilitates self-organization. The conceptual underpinnings of STS [see (Cherns 1987)] have the potential to influence agile practices in several ways. The interested reader may refer to Nerur et al. (2010) for discussion of STS principles as they relate to agile software development.

Drawing from these principles, Nerur and Balijepally (2007) highlight how agile software development teams could be conceived as holographic organizations. These theoretical principles are the basis for our hypotheses formulated in the next section.

**Agile Practices and Project Value**

Agile software development embraces the reality that complexity and change are inevitable when information systems are designed and developed to sense and respond to the demands of a turbulent business environment. Any efforts to control change through elaborate planning exercises would be wasteful and misguided. It is therefore essential to accept and embrace change by adapting the development processes and practices for quick learning and responsiveness. Drawing from the principles enunciated in the agile manifesto, several agile development methods specify different best practices and organizing frameworks for fostering agility in software teams. Adapting from the criteria for agility in software teams articulated by Ambler (2009), we group the processes and practices of agile development into the following categories: stakeholder collaboration, continuous validation/testing, reflection & review, and self-organization/team governance. Unlike Ambler (2009), which considers creating stakeholder value through regular delivery of working software as a criterion for agility, we conceptualize it as an outcome of embracing agile practices and processes. We argue that when software teams institute agile practices and processes covering these four dimensions, they create project value. Figure 1 showcases the research model.

An important tenet of agile development is to get stakeholders actively involved throughout the development process (Cockburn and Highsmith 2001). Where feasible, a collocated user representative, who is available for populating and prioritizing the backlog and for regularly processing requirements for the current iteration, is considered highly valuable. Active stakeholder collaboration in the agile development process is consistent with the holographic principles of ‘building whole into parts’, ‘minimum critical specification’ and ‘learning to learn’. When product owners or user representatives with requisite business knowledge and skills become available to the agile teams on a continuous basis, stakeholders’ goals and priorities at the macro level get translated and integrated into the elemental parts of the system at the micro level. Thus, as the system takes shape it is more likely to satisfy stakeholder interests at multiple levels.
Agile methods also emphasize minimum critical specification of requirements/design and skimp on elaborate requirements gathering and big upfront designs. Implicit in this principle is the assumption that stakeholders or their agents are available to agile teams on a regular basis so that short cycles of requirements specification/design are done for the iteration at hand. Stakeholder collaboration and engagement also fosters team learning through quicker feedback cycles and through dynamic learning processes of perspective making and perspective taking (Boland and Tenkasi 1995). Stakeholder engagement is also likely to curb scope creep, which has been the bane of software and other types of projects. Further, stakeholder participation in evolving test cases and acceptance criteria, particularly in test-driven development, not only helps in managing the expectations of the stakeholders, but also facilitates the delivery of higher business value to them. Thus, active stakeholder collaboration throughout the software development process should contribute to higher project value for the stakeholders. Hence, we hypothesize:

Hypothesis 1: Stakeholder collaboration in agile teams is positively related to the creation of project value

Validation Processes and Project Value

Constant validation of system features and functionality with the desired system requirements throughout the developmental process is an important aspect of agile development. Validation oriented practices such as Test-Driven Development (TDD), multiple levels of system testing, and product demos to stakeholders at various stages of development/version releases are all consistent with the holographic principles of ‘building whole into parts’, ‘requisite variety’, and ‘learning to learn’. Emphasizing multiple levels of testing (e.g., unit tests, regression tests, user acceptance tests, black box vs. white box testing etc.) and doing product demos (e.g., demos of proof of concept/prototypes, demos of pre-release/final versions, etc.) throughout the development process helps validate the system from whole to the part and vice-versa. Frequent demos/reviews not only help in managing customer expectations, but also enable the developers to detect and correct errors early, thus eliminating expensive rework at a later time. The continual cycles of design-code-test-reflect lead to opportunistic designs [see (Guindon 1990)] as developers gain insights from an evolving software product that incrementally adds business value. The combined use of such varied testing and validation processes also minimizes, if not completely eliminates, the chances that the
completed system has major bugs or is lacking in some essential features. In Test Driven Development, where each unit test is designed to first fail and only pass once the feature or functionality is fully implemented, the constant feedback provided by the battery of tests is a great learning tool for agile development teams. The system that finally results from such integrated testing and validation regimen could be expected to contribute to higher project value for the stakeholders. Thus, we predict:

Hypothesis 2: Validation in agile teams is positively related to the creation of project value

**Reflective Improvement and Project Value**

Agile approaches encourage constant reflection and questioning of the underlying assumptions and the current state of affairs throughout the development process (Williams and Cockburn 2003). Agile processes that foster team reflection and learning are quite consistent with the holographic design principles of ‘building whole into parts’, requisite variety, and learning to learn (Morgan and Ramirez 1984). Agile practices such as daily standups, code analysis (both static and dynamic), team reflection sessions conducted at the end of one or more iterations and/or after major releases, post mortem reviews conducted after project completion, and reviews carried out by external auditors, all facilitate ‘reflection in action’ (Schön 1983) and contribute to process improvements and project value. By providing multiple opportunities for reflection from varied sources (e.g., team members, product owners, other project stakeholders etc.), such reviews/reflection sessions provide rich opportunities for learning and course corrections to agile teams. Practices such as paired development as in eXtreme Programming (XP) (Beck 2000), where the work of each developer gets to be constantly reviewed by the coding partner with instant feedback cycles, also helps bring in the benefits of reflective improvements to the level of an individual developer, not just at the level of the project team. It has also been argued that evolutionary delivery of software that includes practices such as self-organization and reflection workshops is conducive to the creation and integration of knowledge (Dissanayake et al. 2013). This leads us to the next hypothesis.

Hypothesis 3: Reflection in agile teams is positively related to the creation of project value

**Self-Organization and Project Value**

Agile development methods place great trust in empowered self-organized teams to come up with innovative solutions to the problems and issues they confront during the systems development process. Many agile practices that contribute to self-organization and empowerment of agile teams are in line with several holographic design principles. The decision latitude provided to agile teams through empowerment is consistent with the principles of minimal critical specification and building whole into the parts. Such empowerment seeks to bring in high level strategic thinking to the level of team members and the product owner. Thus, such self-organizing teams would be able to simultaneously alternate between the strategic intent or the ‘big picture’ and the demands of the specific task at hand. They could learn quickly by doing complex mental simulations of the cause and effects of various decisions/actions and thus make more informed decisions contributing to enhanced project value.

Redundancy of functions is a critical aspect of self-organization, for it would allow developers to assume different roles depending on the demands of the problem at hand. An approach where developers’ roles are interchangeable leads to transfer of knowledge about architectural design, databases, networking, and other aspects of software development. This leads to the next hypothesis:

Hypothesis 4: Self-organization in agile teams is positively related to the creation of project value
Research Method

The research model outlined above is tested using survey data from a secondary source—i.e., a survey of industry practitioners conducted during 2013 by Ambyssoft, Inc. (Ambler 2013). The data included responses of practitioners from across the world. A total of 174 responses were received for the survey. Eliminating the subjects with missing data resulted in 97 usable responses. The IT roles of respondents ranged from developers, architects, quality assurance professionals, business analysts to product owners and project managers. The demographic details of respondents are summarized in Table 1.

<table>
<thead>
<tr>
<th>Demographic Variable</th>
<th>Number of Respondents</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Agile Experience</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Project teams that are agile</td>
<td>47</td>
<td>48.5</td>
</tr>
<tr>
<td>Project teams trying to become more agile</td>
<td>33</td>
<td>34.0</td>
</tr>
<tr>
<td>Currently not on an ‘agile team’, but have been in the past</td>
<td>17</td>
<td>17.5</td>
</tr>
<tr>
<td>Never involved with an agile team</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Position</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Scrum Master/Team Lead</td>
<td>27</td>
<td>27.8</td>
</tr>
<tr>
<td>Agile Team Member</td>
<td>12</td>
<td>12.4</td>
</tr>
<tr>
<td>Architect/Architecture Owner</td>
<td>9</td>
<td>9.3</td>
</tr>
<tr>
<td>IT Manager</td>
<td>9</td>
<td>9.3</td>
</tr>
<tr>
<td>Project Manager</td>
<td>8</td>
<td>8.2</td>
</tr>
<tr>
<td>Programmer</td>
<td>7</td>
<td>7.2</td>
</tr>
<tr>
<td>Product Owner</td>
<td>7</td>
<td>7.2</td>
</tr>
<tr>
<td>Business Analyst</td>
<td>5</td>
<td>5.2</td>
</tr>
<tr>
<td>QA/Testing</td>
<td>3</td>
<td>3.1</td>
</tr>
<tr>
<td>Business Stakeholder</td>
<td>1</td>
<td>1.0</td>
</tr>
<tr>
<td>Others</td>
<td>9</td>
<td>9.3</td>
</tr>
<tr>
<td>Location</td>
<td></td>
<td></td>
</tr>
<tr>
<td>N. America</td>
<td>46</td>
<td>47.4</td>
</tr>
<tr>
<td>Europe</td>
<td>30</td>
<td>30.9</td>
</tr>
<tr>
<td>Australia/New Zealand</td>
<td>7</td>
<td>7.2</td>
</tr>
<tr>
<td>Asia</td>
<td>6</td>
<td>6.2</td>
</tr>
<tr>
<td>South/Central America</td>
<td>4</td>
<td>4.1</td>
</tr>
<tr>
<td>Africa</td>
<td>4</td>
<td>4.1</td>
</tr>
<tr>
<td>Industry Sector</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Technology (incl. software)</td>
<td>38</td>
<td>39.2</td>
</tr>
<tr>
<td>Financial</td>
<td>16</td>
<td>16.5</td>
</tr>
<tr>
<td>IT Consulting</td>
<td>12</td>
<td>12.4</td>
</tr>
<tr>
<td>E-Commerce</td>
<td>10</td>
<td>10.3</td>
</tr>
<tr>
<td>Government</td>
<td>9</td>
<td>9.3</td>
</tr>
<tr>
<td>Manufacturing</td>
<td>1</td>
<td>1.0</td>
</tr>
<tr>
<td>Others</td>
<td>11</td>
<td>11.3</td>
</tr>
<tr>
<td>Department Size (Number of IT/System Development People)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 to 10</td>
<td>21</td>
<td>21.6</td>
</tr>
<tr>
<td>11 to 50</td>
<td>23</td>
<td>23.7</td>
</tr>
<tr>
<td>51 to 100</td>
<td>16</td>
<td>16.5</td>
</tr>
<tr>
<td>101 to 500</td>
<td>23</td>
<td>23.7</td>
</tr>
<tr>
<td>501 to 1000</td>
<td>2</td>
<td>2.1</td>
</tr>
<tr>
<td>1000+</td>
<td>12</td>
<td>12.4%</td>
</tr>
</tbody>
</table>

Table 1 - Demographic Details of Survey Respondents
In this research we conceive project value as the dependent measure that captures the stakeholder value created by the project. The four agility dimensions conceptualized as independent variables in the model capture different dimensions of agility in software development processes—stakeholder collaboration represents the level of stakeholder engagement with the software team throughout the development process; validation dimension captures the level of quality assurance processes embraced by software teams; team reflection dimension signifies the level of reflection and review practices adopted by software teams for seeking continuous improvements; and self-organization dimension embodies the extent of team empowerment and self-governing principles and practices adopted by software teams.

The main survey items that measure the use of various agile practices ask respondents to select the practices that are used in their software development teams. Thus, for each item subject responses were elicited on a binary scale of yes or no categories. For the purpose of this research, for each construct in the model, we grouped relevant items from the survey to create ordinal scales. For instance, for stakeholder collaboration, we first identified 10 items from the survey questionnaire that capture the construct and coded subject responses on an ordinal scale of 0 to 10, based on the number of items selected by the subjects out of the ten items. Using a similar approach, we coded other constructs of the research model into ordinal scales as follows: Validation/Testing – 0 to 10; Reflective Reviews – 0 to 8; Self-Organization – 0 to 8; Project Value – 0 to 6).

We used two control variables in our analysis—respondent’s Agile Experience and Position in their organization. The Agile experience was elicited at 4 levels and coded as an ordinal scale as follows: 1 – Never involved with an agile team; 2 – currently not on an agile team, but have been in the past (and will discuss the most recent one); 3 – currently on a project team where we are trying to become more agile; 4 – currently on a project team where we believe we are agile. To indicate their current positions, subjects were required to select from a list that included 12 categories. As there could be differences in perceptions between respondents in technical positions vs. ones in business/management positions, we grouped respondents into two categories as follows: 1 – Technical Positions (Agile team member, Architect/Architecture owner, Operations/support, Programmer, QA/Testing, and Scrum master/Team lead); 2 – Business/Management Positions (Business Analyst, IT manager, Product owner, Project Manager, Business Stakeholder, Other).

**Data Analysis and Results**

The descriptive statistics and correlations for the various constructs are shown in Table 2. The correlation matrix reveals that project value is positively correlated with all the four independent variables and one of the control variables, i.e., agile experience of respondents. This suggests that respondents with more experience with agile methods perceive higher project value than the ones with lower experience with agile methods. Incidentally, project value is negatively correlated with Position/IT role of the respondents indicating that respondents in technical positions rated project value higher compared to the ones in business/project management positions.

We used hierarchical ordinary least square (OLS) regression to test the main hypotheses of this research. We estimated the model initially (Model 1) using only the two control variables and subsequently added the four independent variables to the base model. The full model can be expressed as follows:

\[
\text{Project Value} = \beta_0 + \beta_1 \cdot \text{AEXP} + \beta_2 \cdot \text{POSN} + \beta_3 \cdot \text{STCO} + \beta_4 \cdot \text{VALN} + \beta_5 \cdot \text{REFL} + \beta_6 \cdot \text{SORG}
\]

where \(\beta_0\) is the constant term; \(\beta_1\) through \(\beta_6\) are the coefficients for the two control variables and the four independent variables of interest: AEXP = Agile Experience; POSN = Position/IT Role; STCO = Stakeholder Collaboration; VALN = Validation; REFL = Reflection; SORG = Self-Organization.
Table 2 – Correlations and Descriptive Statistics

The results of the regression analysis are summarized in Table 3. Model 1 includes only the two control variables—Agile experience and Position. The results indicate that the two control variables are significantly related to project value with the overall model being significant. Model 2 includes the four independent variables of interest in addition to the control variables. The overall model is again significant, and three of the four main effects and the two control variables are significant, as is the change in R² relative to the base model (F change = 11.55; p < 0.01).

<table>
<thead>
<tr>
<th>No.</th>
<th>Variable</th>
<th>Mean</th>
<th>Std. Dev.</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Project Value</td>
<td>3.381</td>
<td>1.365</td>
<td>1.000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Stakeholder Collaboration</td>
<td>5.381</td>
<td>2.089</td>
<td>0.496***</td>
<td>1.000</td>
<td></td>
<td></td>
<td></td>
<td>1.000</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Stakeholder Collaboration</td>
<td>4.526</td>
<td>1.974</td>
<td>0.479***</td>
<td>0.479***</td>
<td>1.000</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Reflection</td>
<td>2.835</td>
<td>1.675</td>
<td>0.429***</td>
<td>0.244***</td>
<td>0.398***</td>
<td>1.000</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Self-Organization</td>
<td>4.165</td>
<td>1.612</td>
<td>0.246***</td>
<td>0.352***</td>
<td>0.428***</td>
<td>0.354***</td>
<td>1.000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Agile Experience</td>
<td>3.309</td>
<td>0.755</td>
<td>0.390***</td>
<td>0.142’</td>
<td>0.316***</td>
<td>0.123</td>
<td>0.095</td>
<td>1.000</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>Position/IT Role</td>
<td>1.608</td>
<td>0.491</td>
<td>-</td>
<td>-</td>
<td>-0.215**</td>
<td>-0.054</td>
<td>-0.077</td>
<td>1.000</td>
<td>0.168</td>
</tr>
</tbody>
</table>

n = 97; *p < 0.10, ** p < 0.05, ***p < 0.01

Table 3 - Regression Analysis Results for the Project Value Dependent Variable

Hypothesis 1 predicted Stakeholder Collaboration in agile teams to be positively related to Project Value. Results from the regression analysis suggest that the coefficient for the main effect of Stakeholder collaboration is positively significant (β₃ = 0.194; p < 0.01) lending support to Hypothesis 1. Hypothesis 2 predicted Validation in agile teams to be positively related to Project Value. From Table 2 it can be seen that the regression coefficient for the main effect of Validation is positive and significant (β₄ = 0.137; p <
Thus, Hypothesis 2 is also supported. Hypothesis 3 predicted that Reflection in agile teams to be positively related to Project Value. Results from the regression analysis (Table 2) suggest that the regression coefficient for the main effect of Reflection to be positively significant ($\beta_5 = 0.218; p < 0.01$). Therefore, Hypothesis 3 is also supported. Hypothesis 4 predicted that Self-Organization in agile teams to be positively related to project value. However, regression results from Table 1 suggest that the coefficient for the main effect of Self-organization is a small negative value and is not significant ($\beta_4 = -0.076; p < 0.30$). Thus, Hypothesis 4 is not supported. The results of hypotheses testing are summarized in Table 4.

<table>
<thead>
<tr>
<th>Hypothesis</th>
<th>Result</th>
<th>t statistic</th>
<th>p-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hypothesis 1: Stakeholder collaboration in agile teams is positively related to the creation of project value</td>
<td>Supported</td>
<td>3.368</td>
<td>&lt;0.01</td>
</tr>
<tr>
<td>Hypothesis 2: Validation in agile teams is positively related to the creation of project value</td>
<td>Supported</td>
<td>2.001</td>
<td>&lt;0.01</td>
</tr>
<tr>
<td>Hypothesis 3: Reflection in agile teams is positively related to the creation of project value</td>
<td>Supported</td>
<td>3.165</td>
<td>&lt;0.01</td>
</tr>
<tr>
<td>Hypothesis 4: Self-organization in agile teams is positively related to the creation of project value</td>
<td>Not Supported</td>
<td>-1.034</td>
<td>0.30</td>
</tr>
</tbody>
</table>

Table 4 - Results of Hypotheses Testing

Discussion

Our study seeks to affirm the claim that stakeholder collaboration, continuous testing and validation, team reflection, and self-organization can lead to project value. As mentioned earlier, our measures were constructed using responses to a survey administered by Ambysoft, Inc. (Ambler 2013). Not surprisingly, all, but one of our hypotheses were supported.

Our results show that stakeholder collaboration is positively associated with project value. The rationale for our hypothesis was that stakeholder engagement would ensure that the evolving code would satisfy acceptance criteria, allow the stakeholders to have realistic expectations of what the software would and would not deliver, provide opportunities for the interchange of valuable domain knowledge, manage the introduction of new requirements later in the lifecycle, and ensure that high business value is delivered early and often. While it may not be possible for stakeholders to be actively engaged throughout the development process, increasing the opportunities to collaborate frequently with the stakeholder can add tremendous value to a project.

Continual testing and validation positively impacts project value. Frequent testing helps in isolating errors early, has the potential to get to the root-cause of the problem, creates options (i.e., opportunistic designs), facilitates learning and exchange of knowledge, and eventually leads to fewer defect densities. Our findings confirm the value that accrues to a project because of continual validation built into the development process.

Our results provide strong support for our hypothesis that team reflection leads to higher project value. ADM affords plenty of opportunities to software teams to assess progress, reflect and introspect on activities to gain an understanding of what works and what doesn’t, and adjust their behaviors in light of knowledge acquired during the process.

Our hypothesis with regard to the positive impact of self-organization on project value was not supported. A plausible explanation for the lack of support is that our contrived measure of self-organization and governance did not adequately capture the richness of the construct. Indeed, the concept of self-organization, which is arguably the crux of ADM, has received scant attention in the literature. Future
research may look at developing a robust measure of self-organization and assessing its effectiveness in delivering project value.

A major limitation of our study relates to the way we tested the model using data from a secondary source. We used data from an existing survey which was compiled for a different purpose—i.e., for understanding the current state of agile implementation in IT organizations. Accordingly, we had to adapt the questionnaire items to the constructs relevant to our study. As most questionnaire items were measured on a binary scale of yes or no responses, we selected items relevant to the constructs in our research model and created variables with ordinal scales. Thus, validity testing using traditional statistical means could not be done. However, we believe the items carry face validity/content validity as these items were used in multiple prior surveys by Ambyssoft, Inc. using industry respondents. Also, the authors evaluated each item by discussing among themselves, and included only the ones that appear to carry face validity and that are found relevant to the constructs of interest here. However, we believe the research model presented here should be validated in a future large sample survey with measures constructed and validated through traditional means.

**Conclusion**

Agile software development methodologies have been proposed and practiced by organizations that develop commercial and non-commercial software for more than a decade. While considerable strides have been made in customizing and implementing agile practices within organizations, much work remains to be done in clearly understanding which practices work best under what circumstances and empirically demonstrating which practices add the most value towards agile development project success. Researchers and practitioners are still trying to clearly delineate the parameters and practices that readily facilitate agility and the different levels of agility that can be achieved. This paper has identified several key dimensions of the agile development process and has investigated the impact of these dimensions on the overall project value. Several hypotheses have been developed based on relevant underlying theory and the research model has been validated using the survey data collected by Scott Ambler (Ambler 2013), one of the leading thinkers of the ADM. The results indicate that stakeholder collaboration is essential for adding value to the project. Similarly, continuous testing and validation contributes to improving project value. Reflective improvement is also a key ingredient for ensuring project value. However, self-organization was not found to have a significant impact on project value. This is contrary to the basic belief within the agile community and the lack of support for this hypothesis could be attributed to the limitations of the data collection and the survey instrument design. However, this does raise the issue of how best to systematically study the impact of self-organization and governance on project value. While the results from this study are preliminary, the findings from the study have generated some useful insights for the successful execution and management of agile projects. Since the data was collected from agile development practitioners, the results provide a glimpse into the state of the development practices and help identify ways to improve the overall agile development process.

Our study provides a good starting point for empirically investigating the value delivered by the many abiding principles that are either explicitly or implicitly present in ADM. Specifically, future research should attempt to evolve richer and more nuanced measures of value drivers such as evolutionary development, process flexibility and/or agility, validation and continuous improvement, and self-organization. The agile literature has long claimed that self-organization is one of the distinctive features of ADM. Yet, a good scale for the construct eludes our grasp. Future research may also be directed towards identifying and validating mediating and moderating factors that impinge on the value that ADM purports to deliver.
Appendix

Survey Questionnaire Items [Adapted from Ambler (2013)]

I. Project Value
What strategies does your team follow to provide value to your stakeholders? Please select that apply (if any)

- We are producing working software every iteration/sprint during construction
- We have a definition of what it means to be done
- We actively consider usability issues in the development of the solution
- We are producing supporting documentation, such as user manuals or operating manuals
- We are implementing improvements to the business process
- We are making business personnel changes as a result of this project

II. Stakeholder Collaboration
What strategies does your team follow to provide value to your stakeholders? Please select all that apply (if any)

- At the start of the project we identified our key stakeholders and their goals
- We have regular discussions with key stakeholders to understand their goals throughout the project

What strategies does your team follow when working with your stakeholders? Please select all that apply (if any)

- Our team has a product owner who represents the stakeholder community
- We work with specific stakeholders, or particularly domain expert, on an as needed basis
- We have access to stakeholders, or their representatives, on a daily basis
- Stakeholders work with business analysts who provide requirements to our team directly
- We have written requirements specification which defines what we need to deliver
- We did some initial requirements envisioning with our stakeholders to identify the scope and to populate our backlog at the start of the project
- Stakeholders work with business analysts who provide requirements to our product owner
- Throughout the project we hold modeling sessions with stakeholder groups to populate the backlog

III. Validation/Testing
What strategies does your team follow to validate their work? Please select all that apply (if any):

- We perform our own regression testing on a regular basis
- We take a TDD approach at the design level (e.g., via xUnit)
- We take a TDD approach at the requirements level (e.g., via acceptance tests or story tests)
- Regression testing is performed by an independent test team in parallel to development
- At the end of the project, “final” testing is performed before releasing the system by an independent test team
- At the end of each iteration we demo our work to key stakeholders
- We have “all hands” demo to a wide range of stakeholders every so often
- We demo the solution to stakeholders every iteration/sprint during construction
- We have a demo sandbox where stakeholders can work with an interim version of the system whenever they like
IV. Reflective Reviews

What strategies does your team follow to improve the way that they work together? Please select all that apply (if any)

- External auditors may review what we are doing during the project to help identify potential improvements
- We hold a post-mortem meeting at the end of each project to identify potential improvements for future project teams
- We hold a retrospective/reflection session several times throughout the project to identify potential improvements for our team
- We had a retrospective/reflection session at the end of each iteration/sprint to identify potential improvements for our team

What strategies does your team follow to validate their work? Please select all that apply (if any).

- We include static code analysis in our build
- We include dynamic code analysis in our build
- We follow non-solo development techniques such as pair programming
- We review our work with other technical people external to the team

V. Self-Organization

What strategies does your team follow to organize how they work together? Please select all that apply (if any)

- Each iteration/sprint we hold a planning meeting where the team determines who will do what that iteration
- The project manager/coach/scrum master assigns tasks to team members
- We hold standup meetings to coordinate our activities
- Our product owner is responsible for prioritizing what our team produces
- We generate reports, such as burn down charts and defect trend charts, manually (perhaps via data entry into a tool)
- We produce a status report at least once an iteration for senior management
- Our tools are instrumented and populate a project reporting dashboard to automatically provide status information
- At least once a week, a senior manager will attend our daily standup meetings to get a status update

References


