2011

CROSS SECTIONAL STUDY OF AGILE SOFTWARE DEVELOPMENT METHODS AND PROJECT PERFORMANCE

Tracy Lambert
Nova Southeastern University, tracy.lambert01@gmail.com

CROSS SECTIONAL STUDY OF AGILE SOFTWARE DEVELOPMENT METHODS AND PROJECT PERFORMANCE

By
Tracy Lambert

A DISSERTATION

Submitted to
H. Wayne Huizenga School of Business and Entrepreneurship
Nova Southeastern University

in partial fulfillment of the requirements
For the degree of

DOCTOR OF BUSINESS ADMINISTRATION

2011

A Dissertation
Entitled

CROSS SECTIONAL STUDY OF AGILE SOFTWARE DEVELOPMENT METHODS AND PROJECT PERFORMANCE

By

Tracy Lambert

We hereby certify that this Dissertation submitted by Tracy Lambert conforms to acceptable standards, and as such is fully adequate in scope and quality. It is therefore approved as the fulfillment of the Dissertation requirements for the Degree of Doctor of Business Administration.

Approved:

Richard V. McCarthy, D.B.A Chairperson	3/21/2011 Date
Leila A. Halawi, D.B.A. Committee Member	4/14/2011 Date
Belay Seyoum, D.C.L. Committee Member	5/02/2011 Date
Suri Weisfeld-Spolter, Ph.D. Acting Chair of Doctoral Programs	5/4/2011 Date
J. Preston Jones, D.B.A. Executive Associate Dean	12 May 2011 Date

H. Wayne Huizenga School of Business and Entrepreneurship
Nova Southeastern University
2011

CERTIFICATION STATEMENT

I hereby certify that this paper constitutes my own product, that where the language of others is set forth, quotation marks so indicate, and that appropriate credit is given where I have used the language, ideas, expressions or writing of another.

Signed: ______________________

Tracy Lambert

ABSTRACT

CROSS SECTIONAL STUDY OF AGILE SOFTWARE DEVELOPMENT METHODS
AND
PROJECT PERFORMANCE

by

Tracy Lambert

Agile software development methods, characterized by delivering customer value via incremental and iterative time-boxed development processes, have moved into the mainstream of the Information Technology (IT) industry. However, despite a growing body of research which suggests that a predictive manufacturing approach, with big up-front specifications, detailed estimates, and speculative plans applicable to manufacturing domains, is the wrong paradigm for software development, many IT organizations are still hesitant to adopt agile approaches (Larman, 2004).

This study extends research based on the Unified Theory of Acceptance and Use of Technology (Venkatesh, Morris, Davis, & Davis, 2003) into the domain of business processes. Specifically, processes related to the ‘behavioral intent’ to adopt agile software development methods. Further, it investigated relationships between adoption and the impact on project performance attributes.

A sample was obtained from a population of IT practitioners from within the IT industry. The sampling frame consisted of members from the global Software Process Improvement Network (SPIN) chapters, Agile User Groups, and I.T. industry conference promoters and presenters. Independent variables included performance expectancy, effort expectancy, social influence, and facilitating conditions, with the dependant variable being behavioral intent to adopt agile software development methods. The independent variable of agile software development adoption and dependent variables of project performance were also included as well as predictive models relating adoption to on-time delivery of project functionality and stakeholder satisfaction.

The variables in the study were measured via a 65-item questionnaire based on previous scales, and tested to ensure validity and reliability. The research questions were developed to identify correlations between performance expectancy, effort expectancy, social influence, facilitating conditions, and the behavioral intent to adopt agile software development methods. Additional questions measured the correlation between adoption and key project performance attributes.

Tracy Lambert

The research found positive correlations between performance expectancy, effort expectancy, social influence, facilitating conditions, and behavioral intent to adopt agile software development methods, positive correlations between adoption and on-time delivery of project functionality and stakeholder satisfaction, and weak positive correlations with post delivery defects and project success rates.

ACKNOWLEDGEMENTS

The completion of a dissertation is not a sole endeavor. It is accomplished with patient support and guidance from many people. While I cannot directly thank all who contributed to this effort, I wish to thank the members of my committee without whom this accomplishment would not have been possible. As my committee chairperson, Dr Richard McCarthy provided me with insightful and timely feedback for all aspects of this effort. I also thank Dr. Leila Halawi and Dr. Belay Seyoum who contributed significantly to this effort with their respective levels of research experience and guidance.

I would also like to thank members of the global Software Process Improvement Network (SPIN) chapters and Agile User Groups who participated in this study by completing the research survey, as well as Eugene Levin, Sudhendu Das, Bob Small, Greg Titus and Wendy Titus for the feedback received in vetting my research survey.

Finally, I offer my sincerest thanks to my family for their continue level of support throughout this journey. In particular, I thank my loving and supportive wife, Debra Lambert, who continued to inspire me with her wisdom and insight and my children Ashley, Brooke, and Zachary Lambert for giving up some of our 'quality time' while I pursued this research.

TABLE OF CONTENTS

Page

Page

LIST OF TABLES

LIST OF FIGURES

Figures

Chapter I

Introduction

Background Information

Incremental and iterative development (IID) approaches, precursors to “modern agile software development methods”, have been in place for several decades now and can be traced back to the 1960’s when first used in the United States air defense SAGE project (Larman, 2004). Additional applications of IID methods were employed on the X-15 hypersonic jet program which laid the groundwork from the National Aeronautical and Space Administration’s (NASA) Mercury, Gemini, and Apollo space programs. The roots of IID can be traced back even further to work done at Western Electric in the 1930’s by Walter Shewhart who proposed a Plan, Do, Study, Act (PDSA) cycle for quality improvement, and was further promoted by W. Edwards Deming in the 1940’s using the PDSA cycle coupled with statistical quality control methods to improve manufacturing processes (Larman, 2004).

The use of IID methods when applied to software intensive projects starts to surface more readily in the early 1970’s with the application of IID approaches used in the United States Trident submarine program, and was further developed under the term “Integration Engineering” at IBM’s Federal Systems Division throughout the 1970s and early 1980s for NASA’s Space Shuttle Flight Software System (Larman, 2004). The use of IID methods for software projects starts to increase in earnest in the early 1980's as both government and private sector software projects start to experience issues with the traditional one-pass, waterfall based approach. To further support the use of IID methods, one only needs to look at the relatively poor success of IT projects in the field over the last few decades (Standish Group, 1994, 1999, 2001, 2003, 2004). As a result, alternative methods to the waterfall approach have continued to evolve. In early 2001 a group of IID researchers and practitioners met in Utah to discuss new methods and practices for improving the success rate of software related projects which resulted in the Agile Manifesto (Cockburn, 2001):

Manifesto for Agile Software Development
We are uncovering better ways of developing
software by doing it and helping others do it.
Through this work we have come to value:

Individuals and interactions over processes and tools,
Working software over comprehensive documentation,
Customer collaboration over contract negotiation,
Responding to change over following a plan.

That is, while there is value in the items on
the right, we value the items on the left more.

(Retrieved from www.agilealliance.org 4/20/2008)

The Agile Manifesto, and associated agile principles, resulted from the culmination of several agile software development methods existing, or under development, at that point in time. These included Extreme Programming (Beck, 1999), Scrum (Schwaber & Beedle, 2002), Dynamic Systems Development Method (Stapleton, 1997), Crystal (Cockburn, 2002), and Feature Driven Development (Palmer & Felsing, 2002). Early users of these methods felt that they positively affected the success rates of software projects (Berinato, 2001) while users of more traditional software development methods felt that agile methods were chaotic and lacked rigor and discipline.

A major difference between these dichotomous views is that traditional methods seek to minimize change over the course of projects through the rigorous use of upfront requirements gathering, analysis, and design approaches, with the intent of gaining higher quality results via a controlled and predictive process, while agile methods acknowledge that change will be inevitable and necessary to achieve innovation through individual initiative (Cockburn & Highsmith, 2001). Agile methods are therefore deemed more adaptable and innovative than the traditional prescriptive and controlling waterfall approach, and while not a panacea for all the challenges identified, agile methods do appear to help teams maintain communication, coordination, and control by selecting a palette of agile methods to help meet the changing needs of software development activities.

While there has been considerable debate about the benefits of agile-based software development methods, such as extreme programming (XP) and agile modeling (AM), there has been little empirical research, beyond case studies and experience reports, to validate many of the practices. For example, there has been considerable research into the XP practice of pair programming, but research into the benefits of agile modeling (AM) is even sparser than that associated with XP (Erickson, Lyytinen, & Siau, 2005). These researchers posit that additional work is needed to investigate claims by many that agile methods are just a repacking of old concepts. As a result, this study extends knowledge into the field of agile software development methods by providing additional insight into the impact that effort expectancy, performance expectancy, social influence, and facilitating conditions have on adopting these methods as well as an understanding of the impact that adoption has on select project performance attributes (on-time delivery commitments, project delivered quality, stakeholder satisfaction, and project success rates).

Statement of Problems

Agile software development methods, characterized by delivering customer value via incremental and iterative time-boxed development processes, have moved into the mainstream of the Information Technology (IT) industry (Cockburn, 2001). However, despite a growing body of research which suggests that a predictive manufacturing approach, with big up-front specifications, detailed estimates, and speculative plans applicable to manufacturing domains, is the wrong paradigm for software development, many IT organizations are still hesitant to adopt agile approaches (Larman, 2004). This hesitation has been primarily attributed to previous experience with traditional software development methods which posit that software intensive projects can be developed in a predictable style which uses a plan-driven, waterfall approach irrespective of the high failure rates of this approach when applied to software development projects (Beck, 1999; Boehm, 2002; Schwaber & Beedle 2002; Cockburn, 2001). Rather, agile methods researchers contend that software development is analogous to new product development and as such, is better served by novel and creative approaches which accommodate high rates of change, is not predicable, and requires adaptive methods to provide competitive advantages for organizations (Larman, 2004).

Purpose of Study

A cross-functional study was conducted to identify factors that can help to mitigate concerns which have been identified by the Information Technology industry when considering the adoption of agile software development methods (Larman, 2004). This study expands upon prior technology acceptance research by extending specific factors of the Unified Theory of Acceptance and Use of Technology (UTAUT) (Venkatesh, Morris, Davis, & Davis, 2003) into the domain of business processes (Venkatesh, 2006) as they relate to the *Behavioral Intent* to adopt agile software development methods, where *Behavioral Intent* is a measure of the likelihood that a person will adopt an innovation where intentions do predict actions (Ajzen & Fishbein, 1980). Additional research was undertaken to investigate the relationship between the adoption of agile software development methods and the impact of adoption on specific IT project performance attributes.

The specific research questions are as follows:

1. Is there a correlation between performance expectancy and the behavioral intent to adopt agile software development methods?
2. Is there a correlation between effort expectancy and the behavioral intent to adopt agile software development methods?
3. Is there a correlation between social influence and the behavioral intent to adopt agile software development methods?
4. Is there a correlation between facilitating conditions and the behavioral intent to adopt agile software development methods?

5. Is there a correlation between agile software development method adoption and project performance attributes of on-time delivery of project functionality, post-delivery defects (quality), stakeholder satisfaction (project team and customer), and project success rates (delivered versus cancelled projects)?

The specific research hypotheses are as follows:

Performance Expectancy

Hypothesis 1

- H_{1o}: Performance expectancy is negatively or not correlated to the adoption of agile software development methods.
- H_{1a}: Performance expectancy is positively correlated to the adoption of agile software development methods.

Effort Expectancy

Hypothesis 2

- H_{2o}: Effort expectancy is negatively or not correlated to the adoption of agile software development methods.
- H_{2a}: Effort expectancy is positively correlated to the adoption of agile software development methods.

Social Influence

Hypothesis 3

- H_{3o}: Social influence is negatively or not correlated to the adoption of agile software development methods.
- H_{3a}: Social influence is positively correlated to the adoption of agile software development methods.

Facilitating Conditions

Hypothesis 4

- H_{4o}: Facilitating conditions are negatively or not correlated to the adoption of agile software development methods.
- H_{4a}: Facilitating conditions are positively correlated to the adoption of agile software development methods.

Project Performance

Hypothesis 5

- H_{5o}: The use of agile software development methods are negatively or not correlated to increases in on-time delivery of project functionality.
- H_{5a}: The use of agile software development methods are positively correlated to increases in on-time delivery of project functionality.

Hypothesis 6

- H_{6o}: The use of agile software development methods are negatively or not correlated to decreases in project post-delivery defects.
- H_{6a}: The use of agile software development methods are positively correlated to decreases in project post-delivery defects.

Hypothesis 7

- H_{7o}: The use of agile software development methods are negatively or not correlated to increases in project stakeholder (project team and customer) satisfaction levels.
- H_{7a}: The use of agile software development methods are positively correlated to increases in project stakeholder (project team and customer) satisfaction levels.

Hypothesis 8

- H_{8o}: The use of agile software development methods are negatively or not correlated to improved success of projects (delivered versus cancelled projects).
- H_{8a}: The use of agile software development methods are positively correlated to improved success of projects (delivered versus cancelled projects).

Demographic information was also captured for future research regarding determinants impacting agile software development method adoption but they were not specifically analyzed in this study. The demographic information collected is as follows:

1. *Organizational culture* to indicate the type of culture in place.
2. *Organizational learning orientation* to indicate single versus double loop learning approaches.
3. *Requirements/feature change accommodation* to indicate the frequency and amount implemented.
4. *Project staffing environment* to indicate volatile or stable environment.
5. *Project team demographics* to indicate years of software development experience, size, and location arrangements.
6. *Incentives* to indicate if individual and/or team incentives were utilized.
7. *Agile practices utilized* to include refactoring, pair programming, test driven development, frequent releases, small iterations, continuous integrations, and others.

Summary

This study focused on the UTAUT factors of performance expectancy, effort expectancy, social influence, and facilitation conditions that identify characteristics which influence the Information Technology professional's 'behavioral intent' to adopt agile software development methods, as well as the impact that adoption has on select project performance attributes (on-time delivery of functionality, post-delivery quality, stakeholder satisfaction, and project success rates). As a result, this study positively contributes to the stream of agile-based software development method adoption research.

The remainder of this study is organized in the following manner:

- Chapter II contains a review of literature encompassing a compilation of applicable Information Technology adoption research, industry sources of agile software development methods, practices, results obtained from using agile software development methods.
- Chapter III describes the research methodology and tools utilized to help in answering the research questions.
- Chapter IV contains the analysis of the data collected using the tools and methods identified in Chapter III.
- Chapter V summaries the results of the study as related to the research questions, identifies limitations of the study, and offers suggestions for future research.

Chapter II

Review of Literature

Introduction

This literature review focused on sources of information that contribute to the identification and use of various elements that were included in the research design and hypotheses that were tested. These sources represent a compilation of applicable theoretical models related to technology adoption, traditional and agile software development methods and practices, and the resource-based view of the firm which examines the link between a firm's internal characteristics and performance in pursuit of sustained competitive advantage (Barney, 1991).

Sources of theoretical models related to technology adoption include the Theory of Reasoned Action (Fishbein & Ajzen, 1975), Theory of Planned Behavior (Ajzen, 1985), Task-Technology Fit (Goodhue & Thompson, 1995), Technology Acceptance Model (Davis, 1989), and the Unified Theory of Acceptance and Use of Technology (Venkatesh, Morris, Davis, & Davis, 2003).

Industry sources of agile software development information include methods such as Crystal Methods (Cockburn, 2002), Dynamic Solutions Development Method (Highsmith, 2002), Extreme Programming (Beck, 1999), Feature Driven Development (Palmer & Felsing, 2002), and SCRUM (Schwaber & Beedle, 2002), as well as applied case studies, industry white papers, and web sites reflecting the use of agile software development methods.

Information Technology Adoption Research

The adoption of Information Technology (IT) has been a topic of considerable research for several decades and has generally focused on theoretical models of technology acceptance (usage) based on behavioral intention to use (Fishbein & Ajzen, 1975), Ajzen (1985), perceived usefulness and ease-of-use (Davis, 1989), and task-technology fit (Goodhue & Thompson, 1995).

The Theory of Reasoned Action

The Theory of Reasoned Action (Fishbein & Ajzen, 1975) posits that individual behavior is driven by behavioral intentions where behavioral intentions are a function of an individual's attitude toward the behavior and subjective norms surrounding the performance of the behavior. Figure 1 is a graphical representation of the Theory of Reasoned Action (TRA).

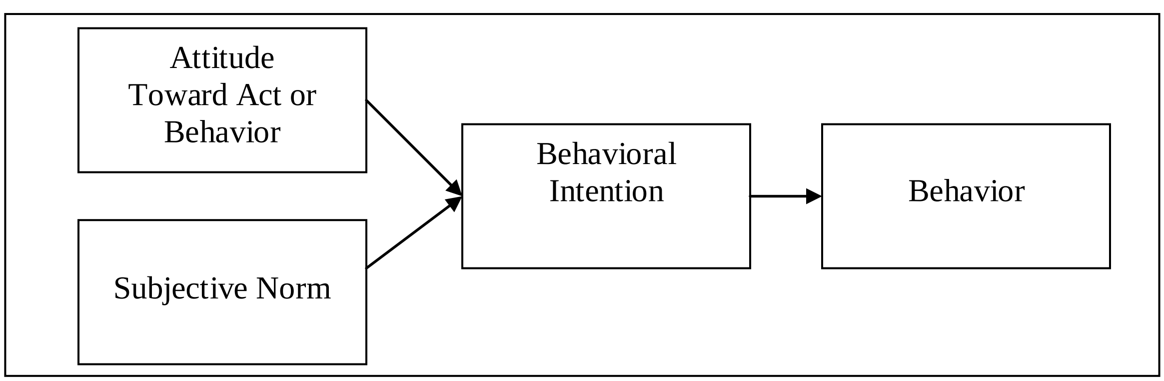

Figure 1. Theory of Reasoned Action (Fishbein & Ajzen, 1975).

The attributes posited in TRA are as follows:

1. *Attitude* which reflects an individual's positive or negative feelings about performing a behavior and is determined through an assessment of one's beliefs regarding the consequences arising from a behavior and an evaluation of the desirability of these consequences.
2. *Subjective norm* which reflects an individual's perception of whether people important to the individual think the behavior should be performed. The contribution of the opinion of any given referent (important other) is weighted by the motivation that an individual has to comply with the wishes of that referent. Hence, overall subjective norm can be expressed as the sum of the individual perception x motivation assessments for all relevant referents.

The Theory of Planned Behavior

The following limiting conditions have been associated with the use of TRA to predict performance (Sheppard, Hartwick, & Warshaw, 1988):

1. *Goals versus behaviors* reflect a distinction between a goal intention and a behavioral intention.
2. *Choice among alternatives* reflects that the presence of choice may change the nature of the intention formation process and the role of intentions in the performance of behavior.
3. *Intentions versus estimates* indicates that there are clearly times when what one intends to do and what one actually expects to do are quite different.

In order to overcome these limitations, Ajzen (1985) developed the Theory of Planned Behavior (TPB) adding *perceived behavioral control* to the model as an additional predictor. This addition accounted for times when people have the intention of carrying out a behavior, but the actual behavior is thwarted because they lack confidence or control over behavior. Figure 2 is a representation of the Theory of Planned Behavior.

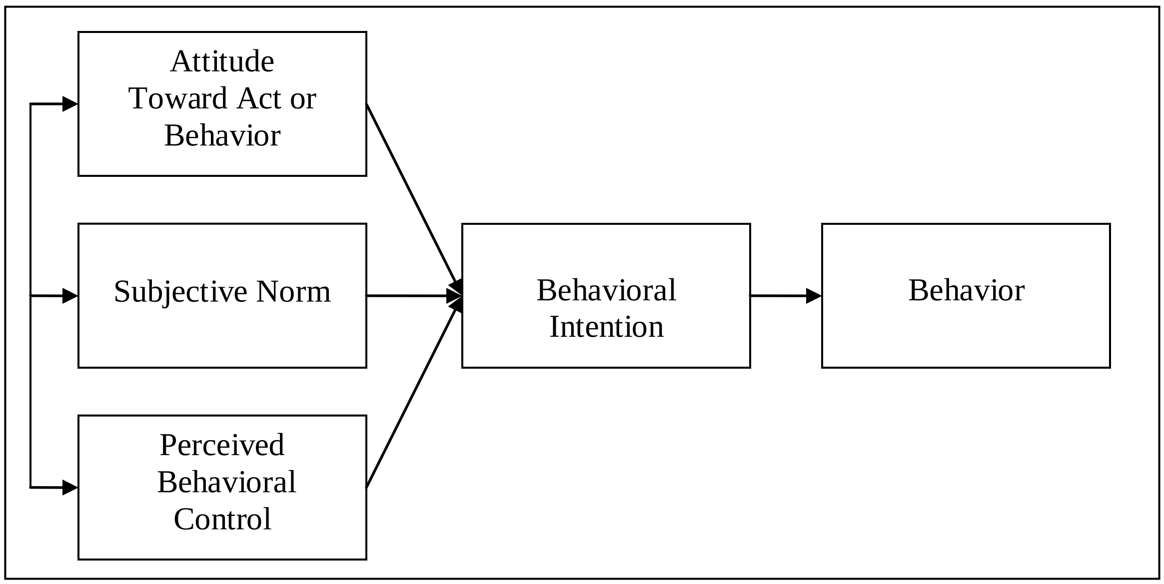

Figure 2. Theory of Planned Behavior (Ajzen, 1985).

Ajzen (1991) noted that the role of perceived behavioral control reflects the concept of *self-efficacy* meaning a conviction that one can successfully execute the behavior required to produce the outcome. Accordingly, perceived behavioral control reflects the perception of the ease or difficulty of the particular behavior and is linked to beliefs about the presence of factors that may facilitate or impede performance.

The Theory of Task-Technology Fit

The Theory of Task-Technology Fit (TTF) has been applied to the acceptance of IT by individuals and posits that IT is more likely to have a positive impact on individual performance and be used if the tasks that the user must perform match the capabilities of the IT (Goodhue and Thompson, 1995). Figure 3 is a graphical representation of the Theory of Task-Technology Fit.

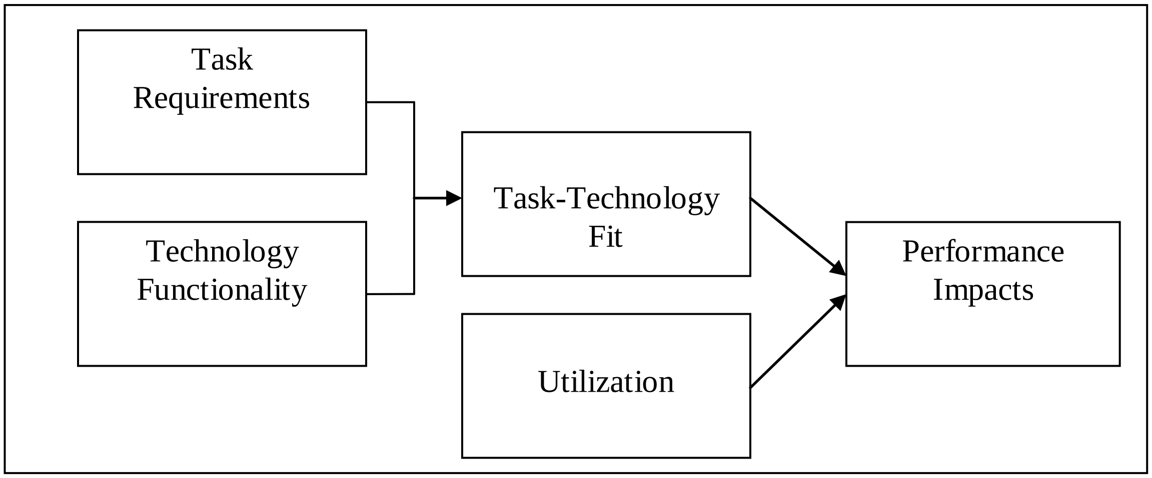

Figure 3. Theory of Task-Technology Fit (Goodhue and Thompson, 1995).

The original TTF included the following factors that were measured using between two and ten questions on a scale ranging from strongly disagree to strongly agree:

1. *Quality* which reflects the currency of data, that the right data is maintained, and with the right level of detail.
2. *Locatability* which reflects how easy is it to find data.
3. *Authorization* which reflects who is authorized to access data.
4. *Compatibility* which reflects compatibility of the data.
5. *Ease-of-use/training* which reflects the relative ease-of-use and training availability.
6. *Production timeliness* which reflects being produced when needed.
7. *Systems reliability* which reflects the reliability of the systems.
8. *Relationship with users* which reflects the level of understanding of business, interest and dedication, responsiveness, delivering agreed-upon solutions, technical and business planning assistance by IT personnel.

TTF Case Studies and Extensions

TTF has been applied in a diverse range of information systems adoption and has been combined with or used as an extension of other models related to IT adoption outcomes as illustrated in the following case studies.

S**upporting Software Maintenance with Software Engineering Tools**

Research conducted by Dishaw and Strong (1998) validated the applicability of TTF in the selection and use of software maintenance tools for improving software maintenance productivity and quality attributes. In their research, Dishaw and Strong noted that the software maintenance process involved two major steps, understanding what production software needed to be changed, and determining how to make the necessary modification. Dishaw and Strong positied that TTF could be used to explain factors which drive or determine usage of software maintenance tools and in so doing would deepen their understanding of the software maintenance process. Their work was predicated on the hypothesis that higher degrees of "fit" expectations between the software maintenance task and maintenance tool functionality would lead to positive consequences of use by individuals choosing to use the technology. Their research confirmed that higher fit between task understanding and modification activity requirements, and software tool production functionality was associated with higher use of tools and that higher fit between task coordination activity requirements and software tool coordination functionality was also associated with higher use of tools.

TTF and Group Support Systems Effectiveness

Research into the application of TTF to determine the effectiveness of using a group support system (GSS) for achieving group tasks has been proposed by Zigurs and Buchland (1998). In their research, Zigurs and Buchland posit an in-depth examination of the combination of complexity's role in tasks and GSS technology issues to gain better insight as to when a GSS is most appropriate for use. The task definition utilized in their research proposal was an extension and refinement of earlier work by Campbell (1988) and Wood (1986) and focused on the central importance of task complexity defined via four dimensions:

1. *Outcome multiplicity* which means that there is more than one desired outcome of a task. An example is where multiple stakeholders have different explicit expectations about what the objectives of a given task are.
2. *Solution scheme multiplicity* which means that there is more than one possible course of action to attain a goal.
3. *Conflicting interdependence* which means that adopting one solution scheme conflicts with adopting another possible solution scheme or when outcomes are in conflict with one another. An example is the classic "quantity versus quality" scenario.
4. *GSS technology* has been defined as systems that combine communications, computer, and decision technologies to support problem formulation and solution generation and include but are not limited to distributed facilities, computer hardware and software, audio and video technology, procedures, methodologies, facilitation, and group data, and as the collective of computer-assisted technologies used to aid

groups in identifying and addressing problems, opportunities, and issues (Zigurs & Buchland, 1998).

The research problems identified by Zigurs and Buckland were indicated in the following proposed “fit profiles”:

1. *Simple tasks* (those with a single desired outcome) should result in the best group performance when done using a GSS configuration that emphasizes communication support.
2. *Problem tasks* (those with a multiple solution scheme requiring the best desired outcome) should result in the best group performance when done using a GSS configuration that emphasizes information processing.
3. *Decision tasks* (those requiring solutions to best satisfy multiple and sometimes conflicting outcomes) should result in the best group performance when done using a GSS configuration that emphasizes processing and process structure.
4. *Judgment tasks* (those requiring solutions when there is conflict and uncertainty in task information) should result in the best group performance when done using a GSS configuration that emphasizes communication support and information processing.
5. *Fuzzy tasks* (those requiring solutions where there is very little focus and group members expend most of their effort on understanding and structuring the problem) should result in the best group performance when done using a GSS configuration that emphasizes communication support and information processing, and includes some process structuring.

As a follow-up to the proposed application of TTF and group support systems, Zigurs, Buchland, Connelly, and Wilson, (1998) performed actual tests of the prescribed fit profiles and confirmed that TTF was generally consistent with the theoretical fit proposed in the fit profiles.

Testing the Technology-to-Performance Chain Model

Research conducted to validate the applicability of TTF when used in both voluntary and involuntary settings was found to have strong support on the impact of performance as well as attitudes and beliefs about use (Staples & Seddon, 2004). The goal of the research conducted by Staples and Seddon was to contribute to the understanding of the predictive validity of TTF. In their research, Staples and Seddon's model tested the following hypothesized precursors of utilization which had not been previously explored regarding TTF:

H_1: TTF will be positively associated with expected consequences of use.

H_2: TTF will be positively associated with affect toward use.

H_3: TTF will be positively associated with performance impacts.

H_4: Expected consequences of use will be positively associated with utilization.

H_5: Affect toward use will be positively associated with utilization.

H_6: Social norms will be positively associated with utilization.

H_7: Facilitation conditions will be positively associated with utilization.

The results of Staples and Seddon's research are reflected as follows:

1. For involuntary use, the model predicted 58% of performance impacts, 64% of expected consequences of use, 41% of affect toward use, and 24% of the utilization construct. Hypotheses 1, 2, 3 and 6 were supported.

2. For voluntary use, the model predicted 48% of performance impacts, 43% of expected consequences of use, 7% of affect toward use, and 17% of the utilization construct. Hypotheses 1, 3, and 4 were supported.

These results reflect strong support for the impact of TTF on performance. The results also reflected that when users do not have a choice about system use, their beliefs about such use may be largely irrelevant in predicting utilization. The study suggest that even in voluntary use settings, a good fit between the task, technology, and user characteristics is very important when the goal is for users to achieve desired performance outcomes from a system.

The Effect of Task and Technology Experience on Maintenance Case Tool Usage

Further research was conducted by Dishaw and Strong (2003) to extend the Maintenance Tool Utilization Model (Dishaw & Strong, 1998) regarding the impact that prior experience has on the use of computer-aided software engineering (CASE) tools for improving software maintainer productivity and quality of maintained software. This research augmented the existing task-technology fit model (Goodhue & Thompson, 1995) with the factors of prior maintenance task experience and prior experience with CASE tools, for explaining tool utilization (Dishaw and Strong, 2003). The hypotheses used in this research are as follows:

H_{1a}: Greater experience with tools is associated with higher use of tools than explained by the Maintenance Tool Utilization Model alone.

H_{1b}: Tool experience interacts with tool functionality in the Maintenance Tool Utilization Model.

H_{2a}: Lower experience with the task is associated with higher use of tools than explained by the Maintenance Tool Utilization Model alone.

H_{2b}: Task experience interacts with task characteristics in the Maintenance Tool Utilization Model.

By exploring the role of task experience in software maintenance tool utilization, Dishaw and Strong (2003) hoped to provide information that would help maintenance managers achieve benefits from the use of tools in their organization. The results of the study confirmed that tool characteristics and tool experience were positively associated with tool use while task experience did not provide significant improvement over the Maintenance Tool Utilization Model. As such, the addition of experience with maintenance CASE tools provides a better explanation of tool utilization than the original Maintenance Tool Utilization Model. Dishaw and Strong thus concluded that the fit between a tool’s functionality and the needs of the task activities, adjusted for the maintainer’s prior experience with the tool, are excellent predictors of a maintainer’s use of a particular tool for a software maintenance project.

Extending Task-Technology Fit With Computer Self-Efficacy

Research to extend TTF to include the construct of *computer self-efficacy* (CSE) on tool utilization indicates that CSE has a direct effect on tool utilization but no significant fit effect (Strong, Dishaw, & Brandy, 2006). In their research, Strong, Dishaw, and Brandy defined computer self-efficacy as a judgment of one’s ability to use a computer and posited that IT utilization in a TTF model is also affected by users’ judgment of their ability to employ computing technology as moderated by the characteristics of the technology being considered. Their results thus add to the continuing body of research extending the application of TTF in the IT domain.

The Technology Acceptance Model

The Technology Acceptance Model or TAM (Davis, 1989) extends previous IT adoption models based on the characteristics of individuals, individual behaviors, and antecedent variables to determine actual use of IT solutions. TAM utilizes two technology acceptance measures to predict technology adoption:

Perceived ease-of-use which reflects the degree to which a person believes that using a particular system would enhance his or her job performance.

Perceived usefulness which reflects the degree to which a person believes that using a particular system would be free of effort.

Davis (1989) posited that a system high in perceived usefulness is one that a user would believe to provide positive performance and that a system perceived to be easy to use is more likely to be accepted by users than one perceived as more difficult to use. Additional research has verified a stronger linkage of perceived usefulness over perceived ease-of-use to actual system usage (Schneberger, Amoroso, & Durfee, 2007). Figure 4 is a representation of the Technology Acceptance Model.

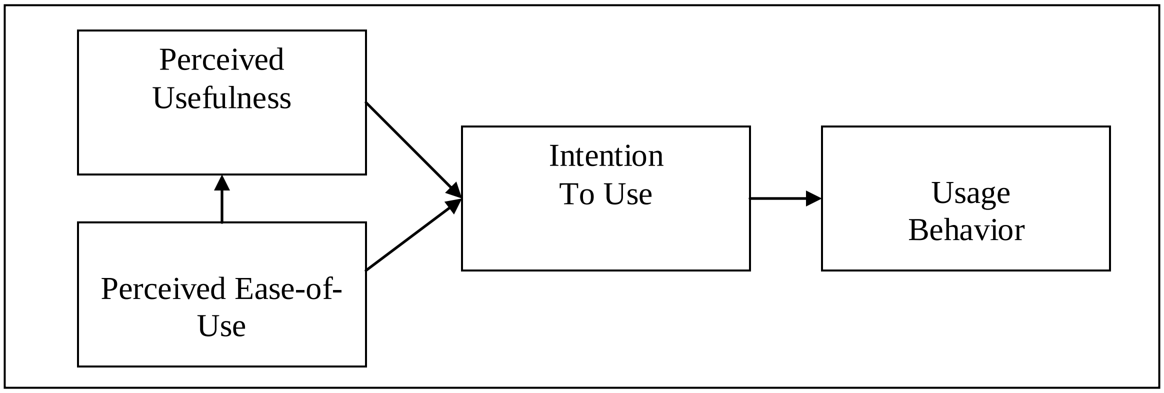

Figure 4. Technology Acceptance Model (Davis, 1989).

TAM Case Studies and Extensions

As illustrated in the following case studies, TAM has been extended and combined with other technology adoption models across as wide range of information technology domains.

Perceive Usefulness, Ease-of-Use, and Usage of Information Technology: A Replication

Shortly after the publication of TAM, additional research was undertaken to validate the psychometrics properties of the ease-of-use and usefulness scales while examining the relationship between ease-of-use, usefulness, and system usage (Adams, Nelson, & Todd, 1992). In research conducted to replicate the previous work by Davis (1989), Adams, Nelson, and Todd conducted two studies to assess the convergent validity and reliability of the TAM scales and found that the TAM scales do demonstrate reliability and validity. Their studies involved the use of electronic and voice mail usage as the dependent variable in study one, and the use of three disparate microcomputer software packages (WordPerfect, Lotus 1-2-3, Harvard Graphics) as the dependant variable in study two. The results of study one confirmed that usefulness was the key determinant of usage. In study two, both ease-of-use and usefulness confirmed the prior work by Davis (1989) regarding reliability and validity of the scales and showed that they can be used to discriminate between software packages. These studies demonstrated that the psychometric properties of the two measures developed by Davis (1989) appeared to be applicable across a wide range of IT adoption scenarios.

Software Evaluation and Choice: Predictive Validation of the Technology Acceptance Instrument

Continuing with research along the lines of TAM scale validation, Szajna (1994) investigated the predictive validity of the ease-of-use and usefulness instrument via a software evaluation and choice scenario which represented a continuation and enhancement of the work begun by Davis (1989). The focus of Szajna's research was to introduce a criterion variable of "choice behavior" rather than "intention to use" as a stronger measure of the subjects' commitment to perform a behavior. The experiment that Szajna conducted used the dependent variable "actual choice" by subjects for a database management system (DBMS) software package that was available for selection from several alternatives available rather than the self-reported "intention-to-use" variable that was present in the original TAM research (Davis, 1989).

The instrument used in the study was a modified version of the original 12-item scale developed by Davis (1989) with wording modified to reflect the use of the DBMS packages. The results of this study further demonstrated that perceived usefulness and perceived ease-of-use are reasonably good predictors of actual choice and that TAM is a logical candidate for use in the evaluation and choice of software packages (Szajna, 1994).

Assessing IT Usage: The Role of Prior Experience

Additional research to determine the predictive behavior of TAM for inexperienced IT users, as well as the identification of the determinants of IT usage for experienced and inexperienced user of a system, was conducted by Taylor and Todd (1995). Their research extended the focused on the role that prior experience plays as a determinant of behavioral intention (Ajzen & Fishbein, 1980) and posited that perceived ease-of-use and perceived usefulness may have different influences depending on prior experience. The view was that users without prior experience would focus on perceived ease-of-use first and while experienced users would focus on perceived usefulness having overcome concerns about ease-of-use (Taylor & Todd, 1995).

The results of the Taylor and Todd study suggested that the augmented TAM, containing the "prior experience" variable, provided an adequate model of IT usage for both experienced and inexperienced users, and could be used to predict subsequent usage behavior *prior* to users having actual use of a system. Taylor and Tood also suggested that this has implications for system design and implementation activities.

Extending the Technology Acceptance Model with Task-Technology Fit Constructs

IT researchers Dishaw and Strong (1998) developed an integrated model of technology adoption utilizing constructs from both TAM and TTF. This integrated IT utilization model provided a better explanation for the variance in IT utilization than either TAM or TTF models alone. The amount of variance in the dependent variable, utilization, explained by this integrated model was 51%, which was higher than the variance accounted for by either TAM or TTF alone. The total effects on utilization were 0.61 for task requirements, 0.37 for usefulness, 0.29 for task-technology fit, 0.26 for ease-of-use, 0.25 for intention to use, 0.18 for attitude toward the tool, and 0.15 for tool experience. In the integrated model, TTF constructs directly affect IT utilization and indirectly affect IT utilization through TAM's primary explanatory variables, perceived usefulness and perceived ease-of-use.

According to Dishaw and Strong, “such understanding is especially important to IT managers who are investing in tools for information users and IT professionals. It should also help tool developers understand how tool characteristics and their fit with task characteristics lead to user choices in respect of using the tool.”

User Acceptance Enablers in Individual Decision Making About Technology: Toward an Integrated Model

Research conducted by Venkatesh, Speier, and Morris (2002) focused on reviewing and reanalyzing data from previous studies on technology adoption (Venkatesh & Speier, 1999) from the standpoint of “user perceptions” for the purpose of developing an integrated model of technology adoption based on the existing TAM (Davis, 1989) and the motivational model (Davis, Bagozzi, & Washaw, 1992). This research (Venkatesh, Speier, & Morris, 2002) examined the influence of pre-training mood and training environment interventions (user acceptance enablers) to understand how user perceptions are formed prior to system implementation.

The integrated model developed by Venkatesh, Speier, and Morris was validated using the following hypotheses:

H_1: Intrinsic motivation will have a significant positive influence on perceived ease-of-use.

H_2: Intrinsic motivation will have a significant positive influence on perceived usefulness.

H_3: Perceived ease-of-use will have a significant positive effect on perceived usefulness.

H_4: Behavioral intention to use a new technology will be determined by intrinsic motivation, perceived usefulness, and perceived ease-of-use.

H_5: An individual's actual technology usage behavior will be determined by behavioral intention to use the technology.

H_6: Continued usage of technology will be predicated by short-term usage behavior.

H_{7a}: User acceptance enablers (pre-training mood and training environment manipulations) will have a positive influence on intrinsic motivation.

H_{7b}: User acceptance enablers (pre-training mood and training environment manipulations) will have a positive influence on perceived ease-of-use.

H_{8a}: The integrated model (excluding user acceptance enablers) will be a better fit than corresponding technology acceptance model.

H_{8b}: The integrated model (excluding user acceptance enablers) will be a better fit than corresponding motivational model.

H_{8c}: The integrated model (including user acceptance enablers) will be a better fit than corresponding technology acceptance model.

H_{8d}: The integrated model (including user acceptance enablers) will be a better fit than corresponding motivational model.

As expected, the integrated model was confirmed by structural equation modeling to be a better fit for predicting user behavior (intention to use) than the existing technology acceptance and motivational models along. By focusing on the key antecedents of usage intention, Venkatesh, Speier, and Morris were able to access the relative degree to which user acceptance enablers, as antecedents to behavioral intention, explained technology acceptance variances. Furthermore, Venkatesh, Speier, and Morris also posited that this research also has practical implications for practitioners' relative to the type of user acceptance enablers (i.e., training interventions) that should be considered as essential components for maximizing technology acceptance.

Extending the Technology Acceptance Model and the Task-Technology Fit Model to Consumer E-Commerce

Additional research by Klopping and McKinney (2004) was conducted in the domain of e-commerce to determine the applicability of an extended TAM/TTF model to predict online shopping activity (both intention to shop and actual purchases). The results indicated that a combined TAM/TTF model was a valuable tool for predicting online shopping activities via e-commerce methods. The combined TAM/TFF model explained 52% of the intention to use variance versus 47% of TAM alone. The results indicated that TAM/TTF models can be extended to other aspects of consumer e-commerce.

An Enhanced Technology Acceptance Model for Web-Based Learning

Research concerning the key determinants of IT adoption in the contemporary education sector indicates that TAM factors are useful in explaining behavioral intention to adopt IT and that computer self-efficacy also has substantial influence on an educator's acceptance of technology (Gong, Xu, & Yu, 2004). In their research, Gong, Xu, and Yu posited an enhanced TAM that included a determinant of computer self-efficacy as a direct antecedent of perceived ease-of-use and intention to use. Their research hypotheses are as follows:

H_1. A teacher's computer self-efficacy has a positive effect on his or her intention to accept web-based learning systems.

H_2. A teacher's computer self-efficacy has a positive effect on his or her perception of ease-of-use about web-based learning systems.

H_{2a}: A teacher's perception on ease-of-use has a positive effect on attitude toward accepting web-based learning system.

H_{2b}: A teacher's perception on ease-of-use has a positive effect on his or her perception on usefulness on the web-based learning system.

H_{3a}: A teacher's perception on usefulness has a positive effect on attitude toward accepting web-based learning system.

H_{3b}: A teacher's perception on usefulness has a positive direct effect on behavioral intention to accept web-based learning system.

H_4: A teacher's attitude has a positive effect on behavioral intention to accept web-based learning system.

Overall results of the study reflected that perceived ease-of-use and perceived usefulness were found to have significant simultaneous effects on a teacher's attitudes. Perceived usefulness had both a direct and indirect effect on intention to use, but the direct effect was more dominant accounting for 41% of the variation on intention versus that of attitude (37%). Computer self-efficacy had a strong direct effect on both perceived ease-of-use and intention to use. According to Gong, Xu, and Yu, the results of their research has significant implications in the real world such that in order for teachers to accept web-based learning systems, it is critical to increase their levels of perceived usefulness and perceived ease-of-use simultaneously. This would translate into systems that have a variety of features to prompt the user's level of perceived usefulness as well as to provide a user-friendly system-human interactive interface to increase their level of perceived ease-of-use.

A Theoretical Integration of User Satisfaction and Technology Acceptance

According to Wixom and Todd (2005), research on user satisfaction of information technology and technology acceptance has been developed in parallel but has not been fully reconciled or integrated. As such, research was conducted by Wixom and Todd for the purpose of reconciling and integrating a proposed research model that distinguishes beliefs and attitudes about a system (i.e., object-based beliefs and attitudes) from beliefs and attitudes about using a system (i.e., behavioral beliefs and attitudes) in order to build the theoretical logic that links the user satisfaction and technology acceptance literature.

The proposed model developed by Wixom and Todd integrates system and design user satisfaction attributes, useful as a diagnostic tool for system development but weak for predicting system usage, with TAM, which is useful for usage prediction but of little value regarding the methods to influence usage through design and implementation. The integrated model was tested using partial lease squares and reflected adequate reliability and convergent and discriminate validity measures and provided preliminary viability for a research model that can differentiate between object-based beliefs and attitudes (system and information quality, system and information satisfaction) and behavior-based beliefs and attitudes (ease-of-use and usefulness, and attitude) when trying to predict usage behaviors. The implications from this research are that managers can have a way to assess system and information characteristics and then reliably investigate their impacts on ultimate usage through the model's proposed causal chain. This also provides a mechanism for understanding and assessing the relative influence of detailed system and information characteristics that can be used to guide system designers when creating systems (Wixom & Todd, 2005).

The Legacy of the Technology Acceptance Model and a Proposal for a Paradigm Shift

While TAM (Davis, 1989), and it's various extensions, have been at the forefront of the technology acceptance research for many years, is has not been without detractors. Richard Bagozzi (2007), who worked on early TAM research (Davis, Bagozzi, & Warshaw, 1989, 1992) presented a critique of TAM identifying several shortcomings and provided for a number of remedies as well as a new perspective on goal-directed behavioral research (Bagozzi, 2007).

Bagozzi notes the following commentary points as fundamental problems with TAM and the current state of the field:

1. *Parsimony* relative to the use of the TAM to determine intentions to use based on perceived usefulness and perceived ease-of-use. This has been the Achilles' heel for TAM as it is unreasonable that one relatively simple model would explain decisions and behavior across a wide range of technologies, adoption situations, and differences in decision making and decision makers.
2. *Links to prior acceptance models*. As with TRA and TPB, researchers, in favor of a simple model, have overlooked essential determinants of decisions and action, and turned a blind eye to inherent limitations of TAM. Researchers have merely attempted to add to TAM rather than deepen TAM in the sense of explaining perceived usefulness and perceived ease-of-use.
3. *Reconceptualizing TAM variables* or adding new ones explaining how the existing variables produce the effects they do. As a result, large gaps exist in TAM between intentions and behavior and between perceived usefulness (PU) and perceived ease-of-use (PEU).

Critical gaps in the TAM framework have been identified by Bagozzi as follows:

1. Validity of the proposed link between intention to use and actual behavior given all the intervening steps and obstacles that may surmount in the time interval between intention and use.
2. Linkage between individual reaction to using information and intentions given that there can be an absence of compelling motivations for acting (i.e., one can accept that PU or attitudes are favorable criteria for deciding to act, but have no desire to actually

act, and could even explicitly decide not to act in the face of extenuation circumstances).

3. The absence of a sound theory and method for identifying determinants of PU and PEU.
4. The neglect of group, social and cultural aspects of decision making,
5. Reliance on naïve and over-simplified notions of affect or emotions.
6. Over dependence on a purely deterministic framework without consideration of self-regulation processes.

Bagozzi (2007) has proposed the following to address the identified problems of motivation content in reasons for acting and how the many reasons are translated into a specific decision to act:

1. Use goal setting, motives, or values to serve as the determinant of decision making rather than the use of attitudes, social norms, and perceive behavioral control.
2. Consider group, cultural, and social aspects of technology acceptance as many decisions with regard to technology acceptance and usage are made collaboratively with others or made with the view of how they fit in or affect people or groups.
3. Understand the effect of emotions on technology acceptance such that a decision maker considers his/her goal and thinks about the aspects of achieving the goal, failing to achieve the goal, and striving to achieve the goals before the adoption decision is made.
4. Self-regulation in the decision making process must be accounted for such that TAM accounts for activation of the will of the agent which operates on deterministic urges

> or desires via reasoning processes rather than cognitive laws of information processing and emotional and motivational laws of responding.

Bagozzi has thus proposed a new foundation for technology adoption/acceptance/rejection that aims for a comprehensive set of core variables and processes that are universal in scope or at least approach universality via the "technology user acceptance decision making core". The approach is defined as follows:

Goal desire -> goal intention -> action desire -> action intention > decision making.

This foundation would be mediated by new developments in psychology and applied related disciplines such that it rests first on specifying fundamental psychological processes of decision making, grounded in universal principles, and second on providing a basis for delineating contingent, contextual causes, and effects of the basic decision making core. This foundation is theorized to result in a deepening of technology acceptance as well as providing additional avenues for better understanding of how, when, and why decisions are made in various technology applications (Bagozzi, 2007).

Looking Forward: Toward an Understanding of the Nature and Definition of IT Acceptance

Additional work by Schwarz and Chin (2007) suggested that the time has come to take a reflective pause regarding the notion of IT acceptance and encourage a view to go beyond the constructs developed in TAM such that a wider understanding of IT acceptance, relative to behavioral usage and its psychological counterparts, can be explored. The perspective offered is that despite over two decades of TAM research, researchers have not explicitly addressed the connection between the general concept of

IT acceptance and IT usage. Schwarz and Chin posit that TAM related research has focused on finding the antecedent factors that most highly relate to the user-based view of IT acceptance and not on a holistic conjunction of a user's behavioral interaction with the IT over time and the psychological understanding/willingness or resistance/acceptance that develops within a specific social/environmental/organizational setting. Schwarz and Chin (2007) suggest that rather than continue to rely on the notion of acceptance-as-extensive usage and chip away at the amount of usage variance explained, researchers may want to consider alternative notions of acceptance where new opportunities may be developed to explore other focal concepts of acceptance.

Schwarz and Chin (2007) also suggest the methodology of etymology (tracing of the history of words) as a starting point to potentially find new factors and concepts not previously uncovered. Their proposed approach includes the following items:

1. Definition of acceptance from the standpoint of four Latin verbs: acceptare, accepto, acceptavi, and acceptatus which essentially equate to receive, to grasp the idea, to assess the worth, to be given, or to submit.
2. Definition of acceptance from the standpoint of the action or the result of the action, describing an aspect of acceptance from the perspective of the passage of time.

The case made by Schwarz and Chin is that these dimensions may prove fruitful in expanding our perspective of IT acceptance when the lifecycle of usage goes beyond initial adoption and includes other goals such as learning, adaption, and optimization.

Unified Theory of Acceptance and Use of Technology

Continuing research in the technology acceptance domain, Venkatesh, Morris, Davis, and Davis (2003) formulated the Unified Theory of Acceptance and Use of Technology (UTAUT) which integrates previous acceptance models and posits the following four direct determinants of usage intention and behavior and up to four moderators of key relationships (gender, age, experience, voluntariness of use):

1. *Performance expectancy* which indicates the degree to which an individual believes that using the system will help him or her to attain gains in job performance.
2. *Effort expectancy* which indicates the degree of ease associated with the use of the system.
3. *Social influence* which indicates the degree to which an individual perceives that important others believe he or she should use the new system.
4. *Facilitating conditions* which indicate the degree to which an individual believes that an organizational and technical infrastructure exists to support use of the system.

UTAUT was developed from the review and consolidation of eight previous models to better explain information systems usage behavior (theory of reasoned action, technology acceptance model, motivational model, theory of planned behavior, a combined theory of planned behavior / technology acceptance model, model of personal computer utilization, innovation diffusion theory, and social cognitive theory).

In testing, UTAUT outperformed each of the individual models with an adjusted R^2 of 69 percent. Figure 5 is a graphical representation of UTAUT.

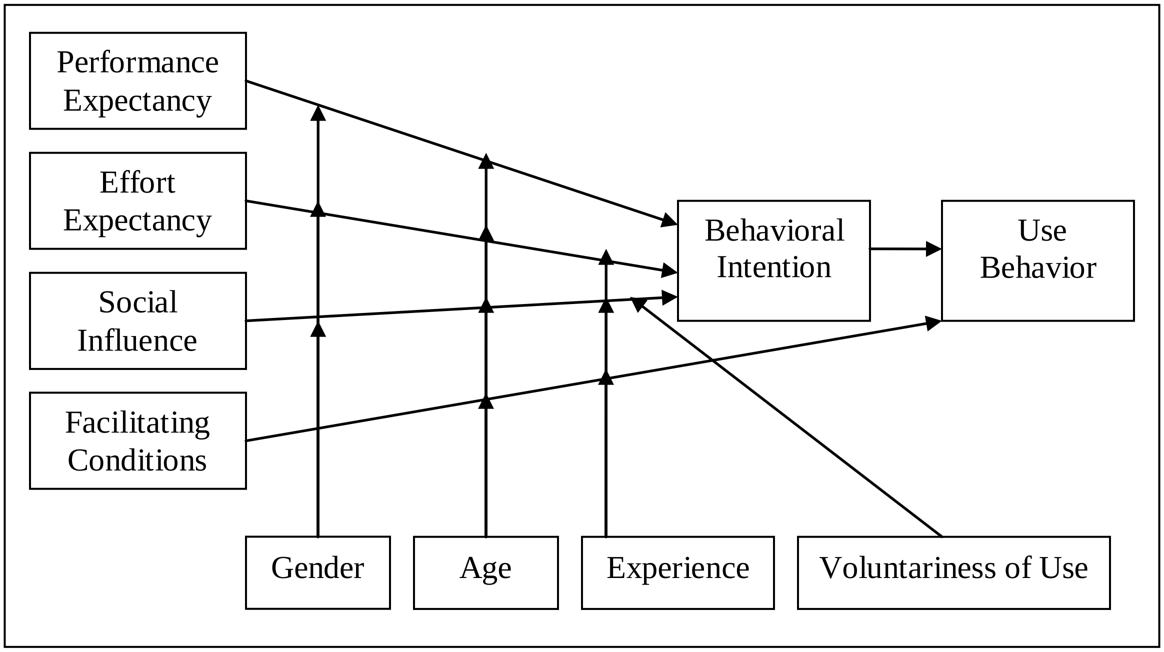

Figure 5. Unified Theory of Acceptance and Use of Technology (Venkatesh, Morris, Davis, & Davis, 2003).

UTAUT Case Studies and Extensions

As illustrated in the following case studies, additional research avenues have been proposed for future directions on technology adoption as well as extending UTAUT beyond the technology domain.

Thoughts on Future Directions for Research on Individual-Level Technology Adoption with a Focus on Decision Making

Venkatesh (2006) put forth the notion that UTAUT could be extended to the following domains as they relate to coordination within firms, across firms, and a firm's interactions with its customers:

1. *Business process change and process standards* which reflect understanding individual adoption of business processes, understanding the impact on employee's

jobs and job outcomes, modeling process characteristics and their impacts on employee's adoption, understanding and isolating change related to technology vs. process characteristics, and interventions to foster success.

2. *Supply-chain technologies* which reflect multiple stakeholder research, broadening the base of constructs, outcomes beyond technology use, and interventions.
3. *Services* which reflect channel choices, service context, and role of technology.

Significant macro-level research on the effects of business process change and process engineering has occurred over the last decade (Grover, Jeong, Kettinger, & Wang, 1995), as well as how IT infrastructure helped to successfully reengineer business processes (Broadbent, Weill, & St. Clair, 1997). However, there has been little focus on individual-level issues with regard to business process standards and process performance, including the impacts on customer satisfaction (Hoogeweegen, Teunissen, Vervest, & Wagenaar, 1999).

Venkatesh (2006) posited that research along the lines of individual adoption of business process could leverage technology-centric determinants identified in prior research and help organizations better predict success of new business processes and create environments that would foster faithful adoption of these processes. These determinants could include the following:

1. Understanding the impact on employees' job and job outcomes.
2. Modeling process characteristics and their impacts on employee adoption.
3. Understanding and isolating change related to technology versus process characteristics.
4. Interventions to foster success.

Venkatesh concluded that while individual-level technology adoption is undoubtedly mature, there are broader areas that can build upon technology adoption research by leveraging robust models available.

The Development, Trajectory and Future of Technology Adoption Research

Venkatesh, Davis and Morris (2007) have posited that although adoption models were originally developed to study technology, they have been extended far beyond their original boundaries to study adoption in such domains as dairy farming and green electricity use. Thus, they put forth the notion that the next step for researchers may be to develop a framework driven set of future research directions that can leverage current knowledge for solving today's relevant business problems.

Venkatesh concluded that while individual-level technology adoption is undoubtedly mature, there are broader areas that can build upon technology adoption research by leveraging robust models available.

Incremental, Iterative, and Agile Software Development Methods

Modern agile software development methods evolved from earlier Incremental and Iterative Development (IID) software development approaches which originated in the 1960's and 70's as part of the United States air defense system projects and American space program (Larman, 2004). As such, IID has a long history of limited use. Fast forward to the late 1990's and we find several modern "agile" software development methods developed by a group of independent IID researchers and practitioners which ultimately culminated in the publication of the Agile Manifesto.

The Agile Manifesto was developed through the amalgamation of several disparate agile methods which were recently developed or under development at that point in time. These methods included the Dynamic Systems Development Method (Stapleton, 1997), Extreme Programming (Beck, 1999), SCRUM (Schwaber & Beedle, 2002), Crystal (Cockburn, 2002), and Feature Driven Development (Palmer & Felsing, 2002). These pioneers attributed their experiences of higher levels of project success rates to the use of agile software development methods when compared to projects developed using traditional (waterfall) software development methods (Berinato, 2001). As a result, acknowledgement of agile software development methods has occurred in the Information Technology (IT) industry but to a somewhat limited degree (Larman, 2004).

Despite some acceptance, contention has developed in the IT industry between “agilest” and “traditionalist” over which software development methods work best. Traditionalist support the notion that the rigorous use of upfront requirements gathering, analysis, and design approaches, via a controlled and predictive process, will lead to better project success rates while agilest acknowledge that change is inevitable and necessary to achieve innovation (Cockburn & Highsmith, 2001). As a result, agile methods have yet to be fully adopted across the global IT industry.

Early Use of IID Methods

The use of IID methods started to increase in earnest in the early-to-mid 1980’s as both government and private sector software projects started to experience issues with the traditional one-pass, waterfall based approach (Larman, 2004)). These failures began to mount so extensively in projects sponsored by the United States Department of Defense

(DoD) that the previous DoD software development standard DOD-STD-2167A, which advocated the use of a waterfall approach, was superseded in December 1994 by a new standard MIL-STD-498 to reflect acceptance of evolutionary requirements and design, and incremental builds of software.

In 2000, the DoD standard 5000.2 was released to further recommend evolutionary delivery and use of IID for software projects (Larman, 2004). This was closely followed by the United States Food and Drug Administration (FDA) which updated their prior waterfall-based development model, FDA97 for FDA approved devices, to FDA02 which promotes the use of iterative development approaches.

Early Pioneers of IID Methods

There are numerous pioneers that advocated iterative methods throughout the 1970s and 1980s (Larman, 2004):

1. Harlan Mills who worked at IBM and promoted iterative development via box-structured methods for systems development with objects.
2. Tom Gilb who developed EVO or evolutionary methods to produce stable requirements.
3. Frederick Brooks who published several books related to IID methods.
4. Barry Boehm who promoted a spiral-development model.
5. James Martin who promoted time-boxing methods for rapid application development (RAD).
6. Tom DeMarco who advocated effective risk management via iterative methods.

7. Ed Yourdon who published numerous articles and books on the topic of iterative development.

Of interest in the literature is the notion that even the man credited with coining the term “waterfall development”, Winston Royce (1970), actually recommended an approach different from what the waterfall method has come to embody. Mr. Royce originally describes the waterfall approach as the most straightforward process, however he did not ascribe to the approach directly opting instead for the use of iterative and evolutionary development methods (Larman & Basili, 2003). Unfortunately, few actually read Royce’s original paper so its iterative connotations were lost to misinterpretation and devolved into the single pass waterfall model known throughout the IT industry today (Larman, 2004).

Modern Agile and Iterative Software Development Methods

Modern agile software development methods include the following:

1. *Adaptive Software Development (ASD)* which was inspired by the Complex Adaptive Systems (CAS) viewpoint (Highsmith, 1999).
2. *Agile Methods* which are characterized by short time-boxed and evolutionary development approaches which emphasize adaptive planning, evolutionary delivery, and include a range of practices and values that encourage rapid and flexible response to change (Larman, 2004). Agile methods promote simplicity, lightness, communication, self-directed teams, programming over documentation, and a low degree of method ceremony (process documentation). A better way to describe agile methods is in terms of ceremony in that they promote “barely sufficient” levels. Agile

methods are adaptive and require feedback to guide project direction. As such, they utilize frequent feedback through early testing and demonstrations of working software.

3. *Agile Modeling* which is not a complete set of processes or methods but rather includes a set of principles and practices for modeling and requirements analysis (Ambler, 2002). In general, agile modeling promotes a low-tech, high-touch collaborative approach to create disposable models to aid understanding and communication and include practices which encourage speed, simplicity, and creative flow (Larman, 2004).
4. *Crystal Methods* which are a family of agile methods that acknowledges iterative development and emphasizes "people" issues over processes (Cockburn, 2002). Crystal methods offer a scaling approach to process ceremony based on project size and criticality which map to an appropriate level of process classification.
5. *Dynamic Systems Delivery Model (DSDM)* which was developed by a group of 16 Rapid Application Development method experts (Stapleton, 1997) and is supported and refined by members of the DSDM consortium. DSDM consists of 3 phases: pre-project phase, project life-cycle phase, and post-project phase. The project life-cycle phase is subdivided into 5 stages: feasibility study, business study, functional model iteration, design and build iteration, and implementation.
6. *Extreme Programming (XP)* which is one of the best known agile methods and emphasizes collaboration, quick and early creation of software, and the use of core support practices (Beck, 1999). These practices include the following:

a. *Planning game* which defines the scope of the next operational release to deliver maximum value to the client.

b. *Small, frequent releases* for evolutionary delivery.

c. *System metaphors* which utilize a simple evocative description of how the program works, such as "this program works like a hive of bees, going out for pollen and bringing it back to the hive" as a description for an agent-based information retrieval system.

d. *Simple design* to avoid design speculation for future changes and focuses on the iteration specific design requirements.

e. *Testing* through unit and acceptance testing via automated test methods where possible. Test first or test driven development approaches are followed.

f. *Frequent refactoring* to simplify and improve design and code as new features are developed. XP emphasizes extensive use of refactoring to “pay down” technical debt which accumulates over the life of a project.

g. *Pair programming* where all code is created by pairs of programmers working at one computer. Pairs rotate frequently between writing and observing code being developed which serves to improve product quality and lower delivery time.

h. *Team code ownership* which emphasizes team ownership of code collectively such that any pair of programmers can improve any code.

i. *Continuous integration* where all code is continuously re-integrated and tested on a separate build machine in an automated fashion on a daily basis.

j. *Sustainable pace* which seeks to avoid frequent, chronic overtime so that developers can enjoy a good quality of life.
k. *Whole team together* where programmers and customers work together in a common project room.
l. *Coding standards* which emphasize the need to have good coding standards that all developers adhere to.

XP is focused on the technical aspects of systems development, proactively responds to change, utilizes good quality practices such as test driven development (TDD),and pair programming, supports the use of open and collaborative communication practices and supports taking an extreme level of activities to deliver a project i.e., extreme testing, extreme code reviews, frequent code integration, extreme customer involvement, short development iterations and frequent feedback via short one to two week iterations if possible (Larman, 2004).

7. *Feature Driven Development (FDD)* which is a development methodology having "just enough" process to ensure scalability and repeatability while encouraging creativity and innovation (Larman, 2004). The principles of FDD are as follows:
   a. A system for building systems is necessary in order to scale to larger projects.
   b. A simple, but well-defined process will work best.
   c. Process steps should be logical and their worth immediately obvious to each team member.
   d. Process pride can keep the real work from happening.
   e. Good processes move to the background so team members can focus on results.

f. Short, iterative, feature-driven life cycles are best.

FDD proceeds to address the items above with a simple process (numbers within the parentheses indicate time spent):

a. Develop an overall model (10 percent initial, 4 percent ongoing).
b. Build a features list (4 percent initial, 1 percent ongoing).
c. Plan by feature (2 percent initial, 2 percent ongoing).
d. Design by feature / Build by feature (77 percent for design and build combined)

8. *SCRUM* which focuses on the management aspects of agile projects and emphasizes a strong promotion of self-directed teams, daily team measurement, and the avoidance of heavy and prescriptive process ceremony (Schwaber & Beedle, 2002). SCRUM includes the following key practices:

a. Self-directed and self-organizing teams.
b. Fixed development iteration lengths (sprints) - 30 days.
c. Fixed iteration content.
d. Demonstrations to clients and associated stakeholders at the end of a sprint.
e. Client driven adaptive planning for each sprint.

A SCRUM sprint is composed of four phases:

1. *Planning* where the project vision, expectations, and funding are secured.
2. *Staging* where requirements (features) are identified and prioritized for the first iteration as well as where initial planning, design, and prototypes occur.
3. *Development* where requirements (features) are identified, prioritized, and estimated for a given sprint along with actual development work (coding and

testing). Feature completion is tracked via daily meetings and burn-down charts which visually reflect feature completion. A review is also held at the end of each sprint to improve process and performance by development teams.

4. *Release* where operational deployment activities occur. Additional activities may include training, marketing, sales and support initiatives.

SCRUM emphasizes a set of project management values and practices rather than requirements analysis, and implementation activities (Larman, 2004). As a result, SCRUM is often combined with other agile practices to form a complementary approach to agile project management.

Increased Use of Modern Agile Software Development (ASD) Methods

The last few years have seen a steady rise in the adoption of modern agile software development methods (Schwaber & Fichera, 2005). However, several industry surveys have also indicated that many organizations are working on co-existence approaches to be able to use both agile and traditional systems development methods in an "ambidextrous" arrangement (Vinekar, Slinkman, & Nerur, 2006). Proponents of this view feel that ASD methods have improved productivity, quality, and customer satisfaction levels where used, but note that other methods are necessary for projects that do not fit the typical mold for IID methods, i.e., project having volatile requirements, unknown technologies, high risk, a need for novelty, or to deliver incremental value. This "incremental value proposition" can be illustrated using the Pareto Principle, also known as the 80-20 rule, where 80% of consequences stem from 20% of the causes. When applied to agile software development methods, customer value is delivered by completing the highest priority customer-defined features or those with high technical complexity or highest risk, first, via time-boxed development cycles called iterations.

This principle is illustrated in figure 6 showing a generalized value proposition of agile software development methods compared to the tradition system development lifecycle (SDLC) waterfall where end-user value is delivered incrementally throughout the life of the project versus a traditional SDLC approach where value is not delivered until the end of a project where months or perhaps even years have elapsed.

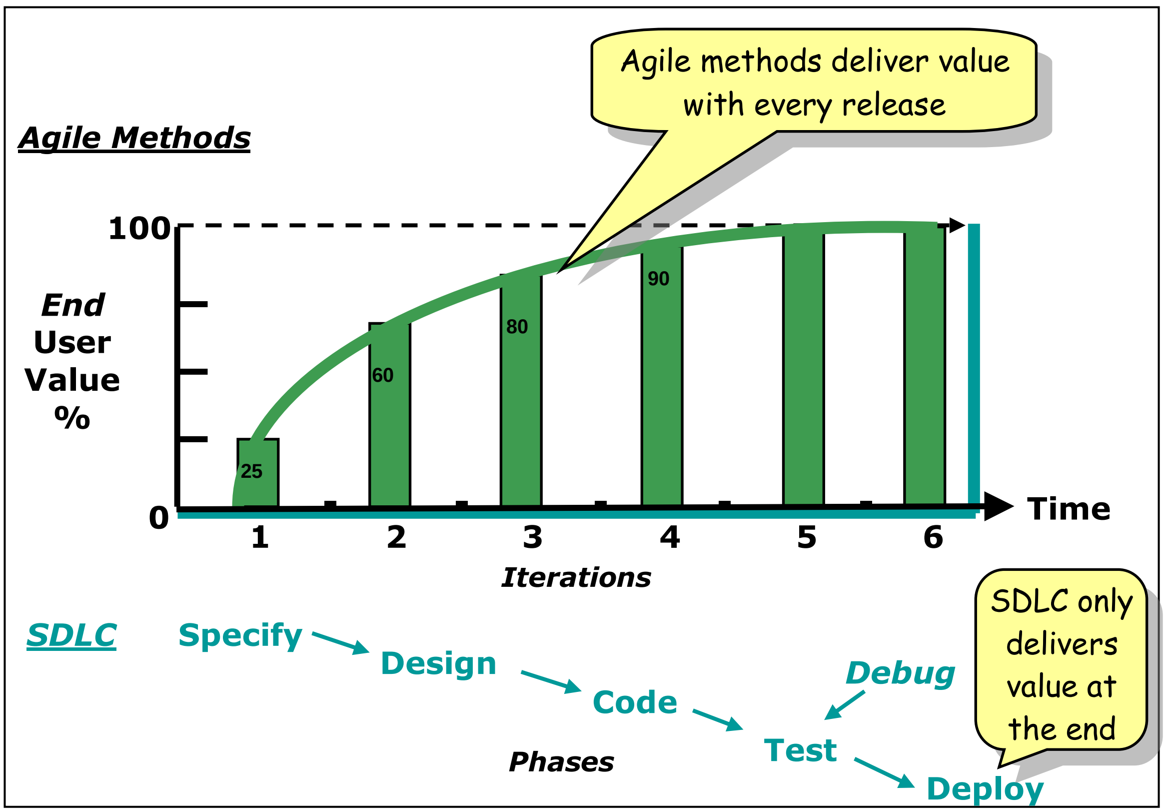

Figure 6. Customer Value Proposition of Agile Vs. Waterfall Methods (McCabe, 2006).

ASD methods deliver functionality (consequences) over several iterations of development activity (causes) so that customer value is gained earlier in the project rather than later. This helps to better manage uncertainty and reduce the negative impact that rework can have on project scope, schedule, and cost (McConnell, 1998).

Agile Software Development Methods Research

While proponents of ASD methods argue that agility provides organizations with adaptable methods, many also contend that stability is still needed for optimization and high assurance of non-agile applicable projects, and as such, systems development organizations need to find middle ground to balance the conflicting interest of stability and agility (Nerur, Mahapatra, & Mangalaraj, 2005).

In support of this duality, researchers have identified four areas where obstacles for establishing ambidextrous approaches might exist: management and organizational, people, process, and technology (Nerur, et al., 2005). For example, the management structure, culture, organization forms, and reward systems of agile and traditional systems development organizations are often at conflict. This level of diversity can cause challenges where organizations, steeped in a hierarchical structure, attempt to adopt agile methods. Organizations also need to review compensation systems to encourage adoption of agile methods so that collective goals outweigh individual successes.

ASD methods also place a premium on people and collaborative interactions. Team roles are interchangeable with team members choosing work assignments regardless of expertise (Martin, 2003). The traditional command and control management structure is largely replaced by self-organizing teams and this approach is sometimes difficult to accept by managers who have previously served as project planner, organizer, and controller. However, this level of flexibility is a key component of ASD methods and a significant factor in differentiating agile from traditional methods.

Proponents of an ambidextrous approach further posit that organizations generally pursue two types of innovation behaviors; exploitation and exploration (March, 1991;

Katila & Ahuja, 2002). Exploitation behavior focuses on items such as core competencies, efficiency, routines, and incremental changes (He & Wong, 2004) whereas exploration behavior focus on items such as experimentation, learning by doing, risk taking, and innovation behaviors.

These behaviors are not mutually exclusive but rather enable organizations to be flexible and innovative without losing the benefits of stability, and efficiency (Katila & Ahuja, 2002). This paradox between exploitation and exploration needs to be accepted and embraced if organizations are to be successful. As such, some researchers posit that organizations which can successfully develop sub-units that are loosely coupled across domains but tightly integrated across the organization, can positively affect overall performance (O'Reilly & Tushman, 2004). This balanced approach suggests that system development organizations can pursue traditional and agile sub-units and thereby reap the benefits of both approaches if a suitable compromise can be reached whereby organizational and management structure, people, process, and technology dimensions can be buffered for each sub-unit yet highly integrated to achieve organizational goals.

Variations in project characteristics can also be a factor in determining the need to balance stability and agility. These factors, which can be used to determine the applicability of traditional or agile development methods, have been identified by Boehm and Turner (2003) as follows:

1. The size of the systems development project and team.
2. The consequence of failure (i.e., criticality).
3. The degree of dynamism or volatility of the environment.
4. The competence of personnel.

5. Compatibility with the prevailing culture.

These factors can then be used to develop a risk profile that can help determine the use of agile or traditional methods as applicable.

Additional research has also been performed relative to how the principles of the Complex Adaptive Systems (CAS) theory (Holmstrom, Fitzgerald, Agerfalk, 2006) can be applied to better understand how ASD methods help meet the challenges of changing business needs. These researchers posit that by mapping agile practices to the CAS principles, and three additional dimensions of process, people, and product, a series of recommended “best practices” can be identified for systems development activities.

As such, early use and success of agile software development methods, primarily for Internet and web-based applications, has sparked interest in using these methods for more mission-critical applications (Glass, 2003; Paulk, 2001). As a result, researchers are seeking ways to better understand how to apply agile methods across a broader range of project types to increase software process and product quality. This is a departure from what has been ascribed to in previous years where a heavyweight, plan-driven approach was deemed best for managing and developing the somewhat chaotic process of software development. To this end, the use of agile processes, which are based on incremental, cooperative, straightforward, and adaptive practices, can be shown to align with the following principles of CAS:

1. *Open Systems* interact with their environment to exchange energy or information within its environment and operates at conditions far from equilibrium.
2. *Interactions and relationships* reflect dynamic exchange of energy or information with each other.

3. *Transformative feedback loops* are direct and indirect transformations feedback loops developed across boundaries.
4. *Emergent behavior* is not predictable given the rich, dynamic, and non-linear type of interactions so prediction is not viable.
5. *Distributed control* where control is distributed rather than concentrated.
6. *Shadow structure* where a minimal structure is needed.
7. *Growth and evolution* reflects continuous growth and evolution enhance survival as systems respond to emerging internal and external environment changes.

When CAS principles are applied to the following ASD practices we find that a shift toward agile methods is informed by these principles:

1. *Frequent release and continuous integration* where frequent releases are critical to accommodate changing requirements. The CAS principle of growth and evolution emphasizes providing a background for such behavior. This supports the use of agile best practices of developing the information system solution iteratively, with a simple set of development processes and tools that are modified over successive iterations, and to start with a small development team and scale up as the project increases in size and complexity.
2. *Need for feedback* with development teams establishing transforming feedback loops across all stakeholders to create an adaptive development environment. The CAS principle of transformative feedback loops emphasizes the importance of feedback in the people, product, and process dimensions provide a background for such behavior. This supports the use of agile best practices to test and validate the information system (IS) solution in each development iteration for the purpose of obtaining

feedback and to make modifications going forward, use time-boxed development cycles and track process milestones within iterations, and involve stakeholders and fellow developers in a pragmatic way by seeking and listening to their comments and concerns.

3. *Proactive handling of changes to the project requirements* where agile methods embrace changes in requirements and underscores a willingness to accept change as an inevitable part of systems development. The CAS principle of emergent order helps explain the phenomenon of unanticipated requirements and adaptive approaches in the IS solution. This supports the use of agile best practices to allow for flexibility in the development process, allow for teams to respond quickly to local needs, and to accommodate changes to requirements as they emerge in the course of changing business needs.

4. *Loosely controlled development environment* where agile development teams are characterized as being flexible and have a distributed control structure (Rihania & Goyer, 2001). The CAS principle of distributed control emphasizes that when there are conditions of high uncertainty, flexibility and adaptability are more appropriate than rigid and static ones. Moreover, project managers of agile teams should help the team be more productive by offering suggestions about how things can be done rather than mandating them (Fowler, 2002). This supports the use of agile best practices for leadership and decision making to be decentralized with more decision making made at local levels, for successive iterations to be fairly independent or loosely coupled, and that the IS solution needs to be componentized with loose coupling and high cohesion within the finished product.

5. *Planning kept to a minimum* where on the surface, agile methods appear to lack planning, however planning is done at the project, iteration, and daily work activity levels emphasizing a “barely sufficient” approach so as to not plan beyond a point of practicality. The CAS principle of emergent order emphasizes accounting for minimal planning to accommodate various unforeseen requirements changes. This supports the use of agile best practices that planning for IS solutions, processes actually used, and the development team structure and composition is best addressed within each iteration.
6. *Enhancing continuous learning and continuous improvement* as agile methods place a great level of emphasis on people and their talents, skills, and knowledge which suggests that the most effective teams are responsive, competent, and collaborative (Boehm, 2002; Cockburn & Highsmith, 2001). The CAS principle of growth and evolution provides support for this behavior. This supports the use of agile best practices that foster reuse and learning from past experiences, experimentation to achieve desired results, and interactions among development team and stakeholders.
7. *Emphasis on working software product* where agile approaches stress a minimalist view of documentation to accommodate change and to reduce the cost of moving information between people (Cockburn & Highsmith, 2001). They also emphasize working code as the benchmark for easier software maintenance (Grenning, 2001). While not directly supported by a CAS principle, the theory of the “path of least effort” (Zipf, 1949) can help explain the emphasis on working product over voluminous documentation. This supports the use of agile best practices that emphasis simple and efficient delivery of error-free functioning solutions, enable the

development process to become most effective at producing solutions, and encourage strong relationships of development team members to produce natural configuration that foster rapid production of working solutions.

The CAS theory therefore provides for a theoretical underpinning of how ASD methods provide teams with the ability to develop highly evolvable and responsive software solutions given the interplay between the people, process, and product dimensions of information systems.

Additional research has shown that agile methods can also help successfully reduce the impact of distance (temporal, geographical, and social-cultural) on global software development (GSD) activities (Holmstrom, Fitzgerald, & Agerfalk, 2006). For example, many multinational enterprises now engage in GSD to exploit new market opportunities by creating virtual corporations and teams which use a "follow the sun" approach for software development (Herbsleb & Moitra, 2001). However, while GSD can provide strategic advantages, it also presents challenges that co-located teams do not encounter. These challenges include physical separation of project teams and clients, which can result in resistance due to perceived job displacement, loss of control and governance, and issues with different communication styles. As a result, while many organizations are engaging ASD methods, others are still hesitant to do so. Ultimately however, the trend in GSD is increasing for three important reasons (Herbsleb & Moitra, 2001):

1. Business advantages of being close to new markets and customer information.
2. Exploiting market opportunities via quick turn-around time of new development opportunities.
3. Flexibility to respond to merger and acquisition opportunities.

As a result, software development via multisite, multicultural, globally distributed arrangements is increasing and in order to capitalize on opportunities, organizations are using ASD methods to reduce the impact of temporal, geographical, and social-cultural challenges through the use various agile practices. These practices include distributed extreme programming, pair programming, small releases, simple designs, testing, refactoring, collective ownership, 40 hour work week, and coding standards, which can be attributed to most of the practices found in Extreme Programming and SCRUM agile methods (Fitzgerald, Harnett, & Conboy, 2006), and while they can help project teams overcome certain constraints associated with traditional software development methods when applied to GSD, the main challenges lie with complexities associated with maintaining good communication, coordination, and control with dispersed project teams (Agerfalk, 2004).

Differences in temporal distance (different time zones) require flexibility so agile-based teams often overlap certain hours in the work day to be able to maintain awareness of time-critical activities. The major challenge in geographic distance is helping project teams maintain a sense of “teamness”, so agile-based teams use daily stand-up meetings to help maintain team cohesion. Finally, social-cultural distance is generally manifested in language differences, so agile-based teams use informal communication methods to develop better relationships and increase the flow of information (Kotlarsky & Oshri, 2005). While not a panacea for all the challenges identified, agile methods do appear to help teams maintain communication, coordination, and control by selecting a palette of ASD methods to help meet the needs of global software development activities.

There has also been continued interest in understanding the link between the effect of organization form and the ability of an organization to develop agile adoption practices (Hovorka & Larsen, 2006). For example, network organizations are distinguished by flexibility, decentralized planning and control, and lateral ties with a high degree of integration of multiple types of socially important relationships across formal boundaries (Van Alstyne, 1997). Previous innovation diffusion research (Roger, 1995) has focused on how communication channels and opinion leaders shape adoption but not on the impact that network mechanisms have on adoption. However, Hovorka and Larsen (2006) proposed that network processes, and theories about knowledge acquisition and adsorption, better allow for organizations to detect and seize opportunities for IT innovation. These theories include the characteristics of communications networks, social information processing, homophily (selection of others similar to oneself) and absorptive capacity. Hovorka and Larsen also posited the notion that social communication networks are relevant for an organizations ability to acquire knowledge about new IT solutions, and that homophily can influence the formation of these communication networks thereby improving knowledge acquisition and utilization.

Absorption capabilities, the set of abilities to manage internal and external knowledge, can also be used to facilitate knowledge transfer through four dimensions (Zahra & George, 2003): acquisition, assimilation, transformation, and exploitation. These dimensions are linked through social integration mechanisms and can improve an organization absorptive capacity for sharing knowledge and rewarding knowledge transfer. In their research, Hovorka and Larsen (2006) found that the processes that occur within social networks can increase agile adoption practices, that network formation is influenced by the perception of similarities between members of the network, in particular, between lead organizations and other consortia-based network members, and that dynamic absorptive capacity at the network level can be enhanced through strong network ties and through social information processing of positive, supporting information. They describe an agile adoption practices model that proposes interactions within the inter-organizational network that enables agile adoption via dependent practices rather than viewing them as separate, independent processes. As a result, they posit that organizational agility can be enhanced if network communication theories are considered together rather than separately and that doing so increases the agility of an organization to adopt IT-based innovations.

Systems Development Method Selection

System development methodologies have continued to evolve as evidenced by continued research into method engineering (Siau, 1999). As a result, there is a plethora of system development methods available. However, the high failure rates of systems development efforts (Hirsch, 2002) and the complexity of many traditional system development methods, suggest that choosing the right method is a complex and difficult task. As a result, the notion of theoretical and practical complexity has been introduced which suggests that a one-size-fits-all software development process does not fit the often turbulent nature of software development (Erickson & Siau, 2003; Erickson, Lyytinen, & Siau, 2005), and that agile methods would appear to be a better fit than traditional methods. These researchers posit that additional work is needed to examine the purported benefits of agile methods, beyond the XP practice of pair programming, and agile modeling, to fill the gaps in the literature regarding potential benefits of agile methods.

Agile Software Development Results

A number of industry surveys have been recently conducted which reflect the relative level of penetration agile software development methods across the IT industry as well as the results received from using these methods.

Scott Ambler Agile Surveys

Reviews of agile software development surveys, conducted in 2006 which sampled 4232 IT industry respondents (Ambler, 2006), and again in 2008, which sampled 642 IT industry respondents (Ambler, 2008), reflected the results listed in Figures 7, 8, 9, and 10:

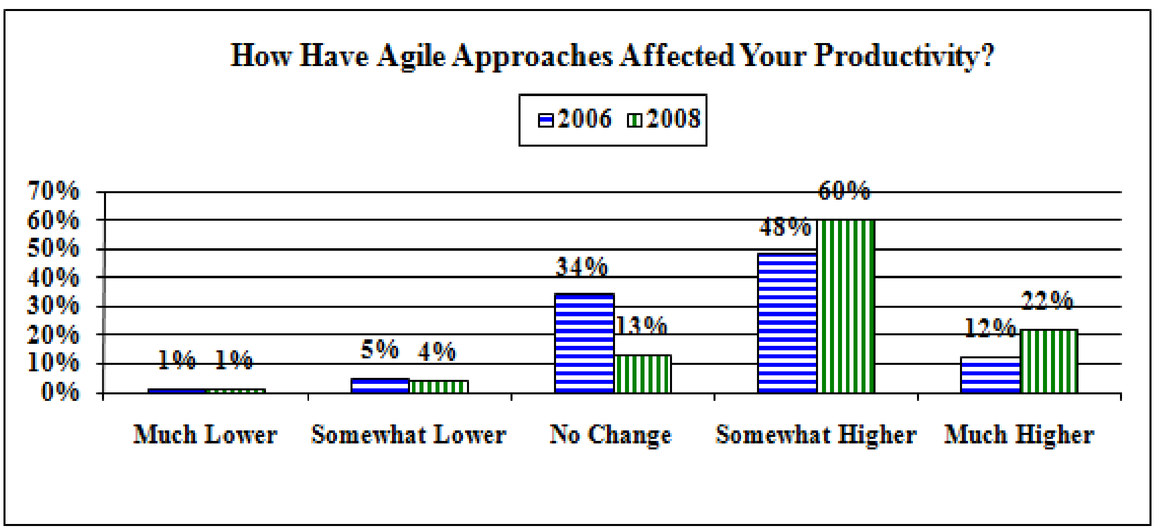

Figure 7. Productivity Results (Ambler, 2006, 2008).

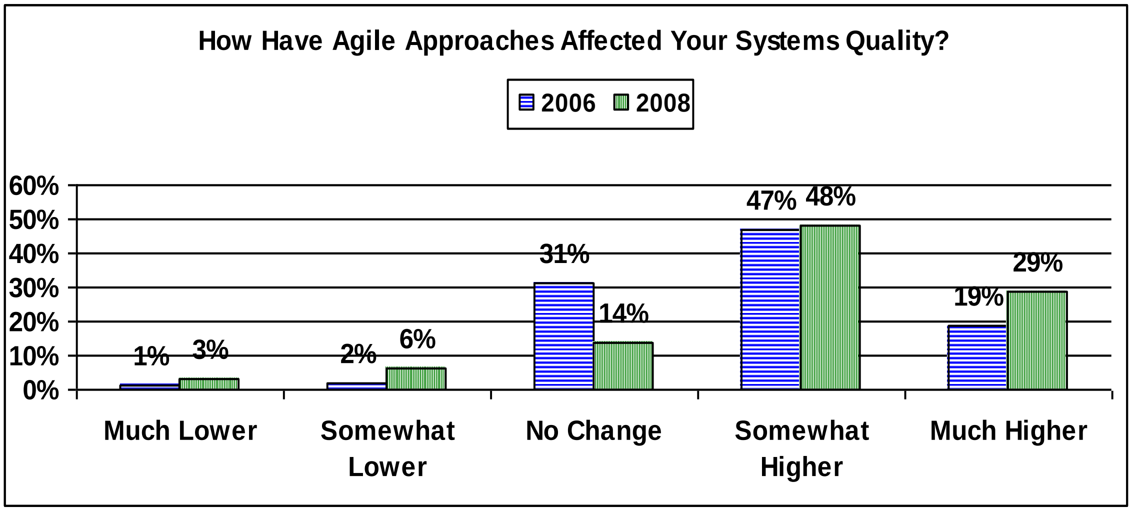

Figure 8. Quality Results (Ambler, 2006, 2008).

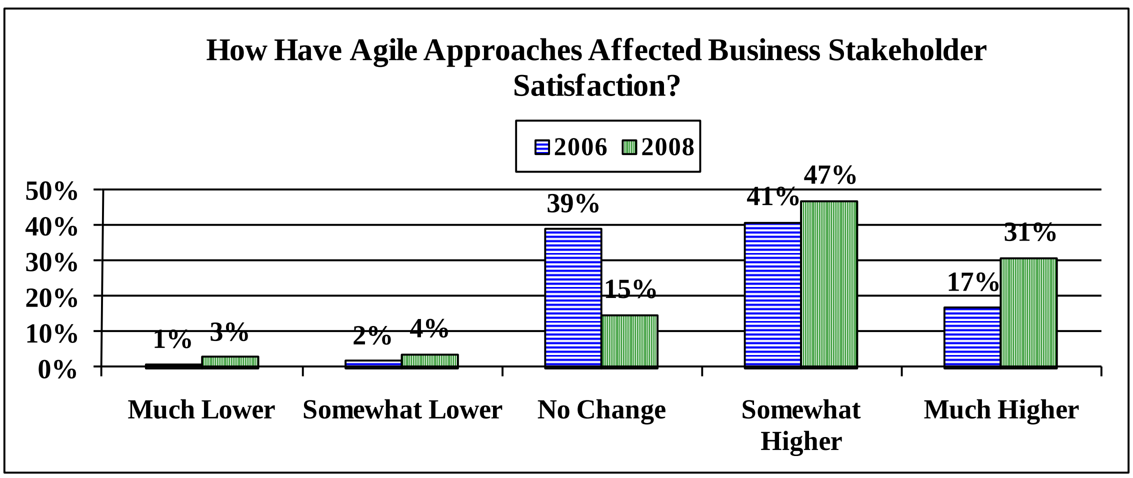

Figure 9. Business Stakeholder Satisfaction Results (Ambler, 2006, 2008).

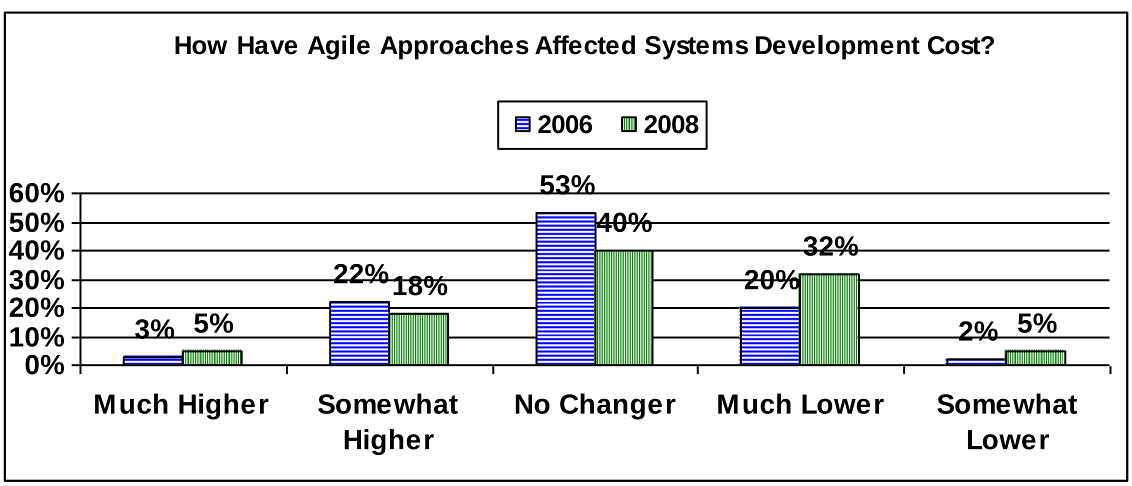

Figure 10. Systems Development Cost Results (Ambler, 2006, 2008).

Agile Project Leadership and VersionOne Surveys

Additional agile software development surveys conducted jointly by the Agile Project Leadership network and VersionOne in 2007, which sampled 1700 IT industry respondents in 71 countries, and again in 2008, which sampled 3061 IT industry respondents in 80 countries (VersionOne, 2008), reflected the results listed in figures 11, 12, 13, and 14:

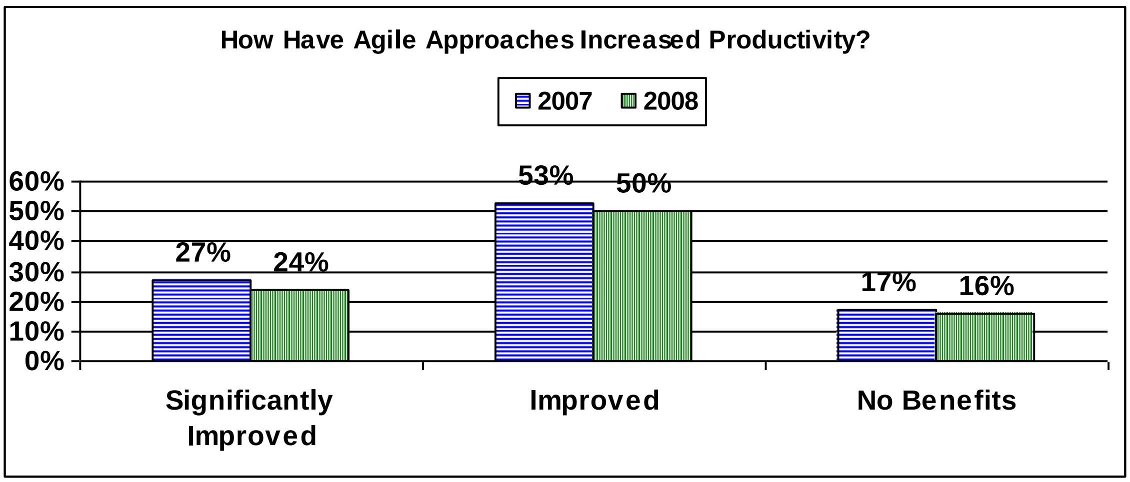

Figure 11. Productivity Results (VersionOne, 2007, 2008).

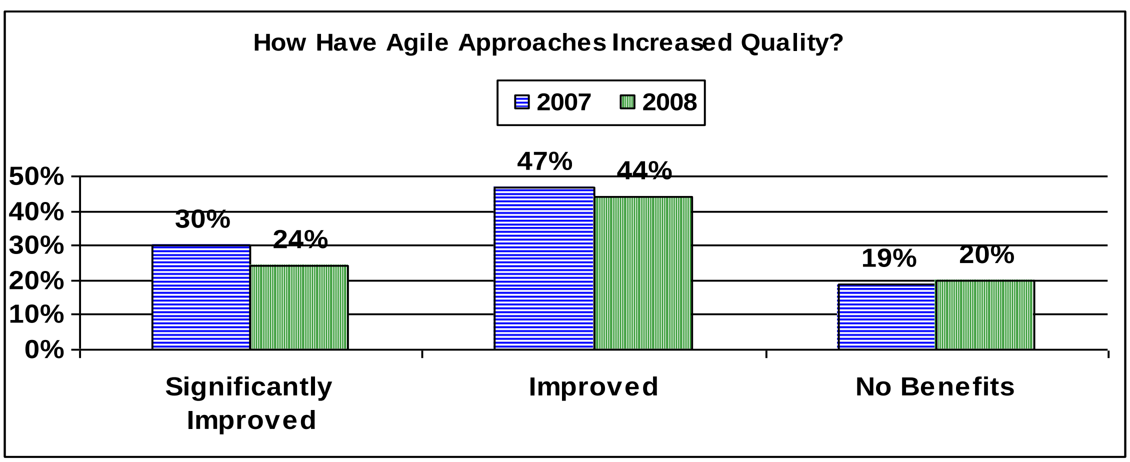

Figure 12. Quality Results (VersionOne, 2007, 2008).

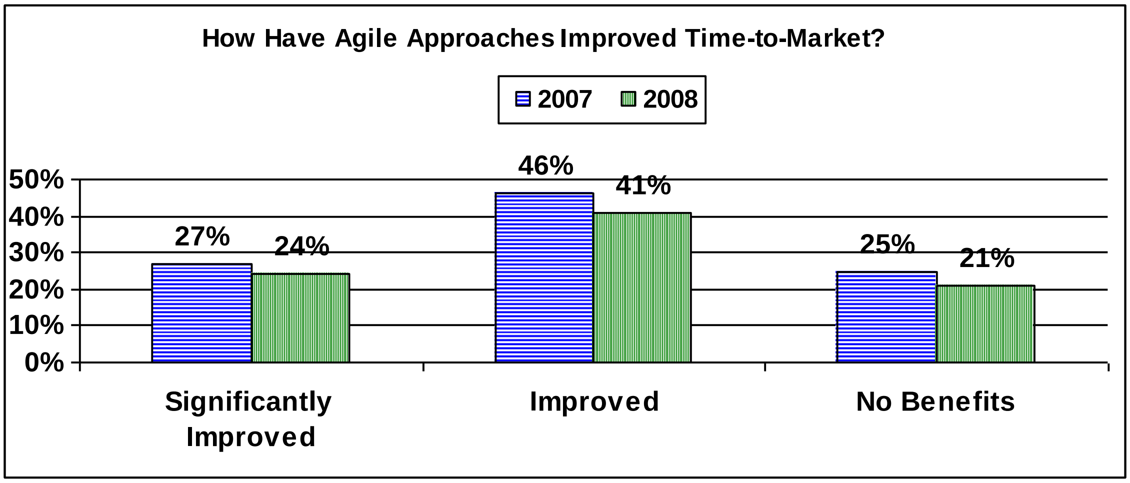

Figure 13. Time-to-Market Results (VersionOne, 2007, 2008).

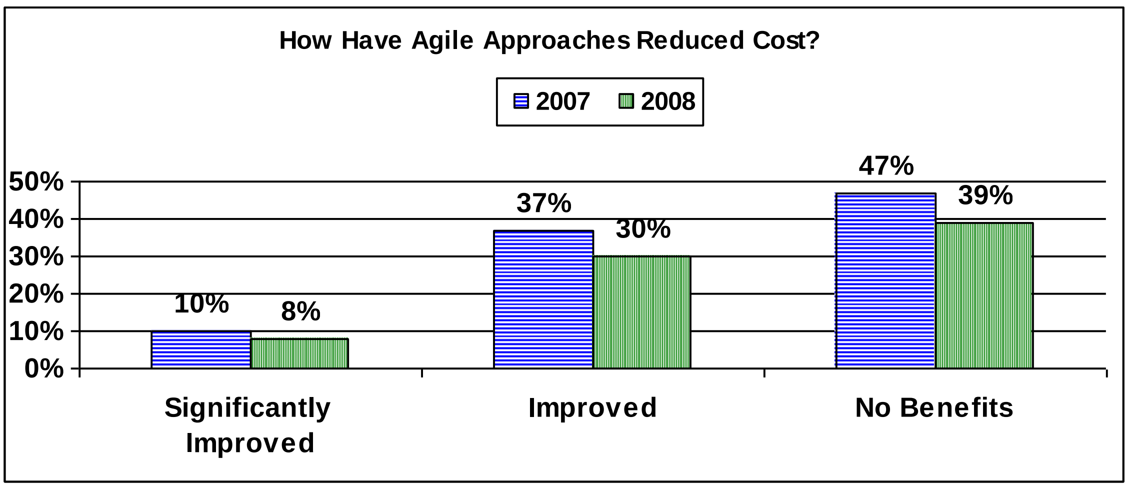

Figure 14. Cost Results (VersionOne, 2007, 2008).

Collectively, these surveys reflect positive improvements for organizations adopting agile software development methods in the areas of productivity, delivered system quality, business stakeholder satisfaction, and reduced development cost.

Firm Resources and Sustained Competitive Advantage

The Resource Based Viewed (RBV) of The Firm (Barney, 1991) was developed to identify sources of sustained competitive advantage for firms based on resource advantages that may be available to a firm. As such, RBV can serve as a basis for understanding the link between the use of agile software development methods and performance.

RBV was developed to extend the level of understanding of the sources of competitive advantage for firms by moving beyond the work done by Porter and his colleagues (Caves & Porter, 1977; Porter, 1980, 1985). Their work primarily focused on both internal analysis of organizational strengths and weaknesses and external analyses of opportunities and threats to describe the environmental conditions that favor high levels of firm performance. This type of research placed little emphasis on the idiosyncratic firm attributes that can contribute to a competitive position by adopting two simplifying assumptions (Porter, 1990):

1. Firms with an industry or strategic group are identical in terms of the strategically relevant resources they control and the strategies they pursue (Porter, 1981).
2. Resource heterogeneity develops in an industry or group (perhaps through new entry) and that this heterogeneity will be short lived because the resources that firms use to implement their strategies are highly mobile (Barney, 1986; Hirshleifer, 1980),

RBV seeks to understand and explain the link between a firm's internal characteristics and performance, and therefore cannot build on these assumptions. As such, Barney (1991) substitutes two alternate assumptions when analyzing sources of competitive advantage:

1. Firms in an industry or group may be heterogeneous with respect to the strategic resources they control.
2. These resources may not be perfectly mobile across firms and thus heterogeneity can be long lasting.

RBV classifies resources in three categories as follows:

1. Physical Capital – Tangible (Williamson, 1975) – including physical technology, plant and physical equipment, geographic location, access to raw materials.
2. Human Capital – Intangible (Becker, 1964) – including the training, experience, judgment, intelligence, relationships, and insight of individual managers and workers in a firm.
3. Organizational Capital – Capabilities (Tomer, 1987) – including a firm's formal and informal reporting and planning structure, controlling and coordinating systems, and interpersonal internal and external relationships.

Barney uses RBV to put forth the notion that a firm can sustain a competitive advantage when implementing a strategy not simultaneously being implemented by any of its current or potential competitors (Barney, McWilliams & Turk, 1989) and the advantage continues to exist after efforts to duplicate that advantage have been created irrespective of calendar time, although it will not last indefinitely (Lippman and Rumelt, 1982).

RBV further extends the work of Porter, who introduced the value chain concept to assist managers in isolating potential resource-based advantages for their firms (Porter, 1985), by examining the attributes that resources must possess in order to be considered sources of sustained competitive advantage. These attributes are as follows:

1. *Value* in that a resource must be valuable in the sense that it exploits opportunities and/or neutralizes threats in a firms' environment.
2. *Rare* in that a resource must be rare among a firm's current and potential competitors.
3. *Imitable* in that a resource must be imperfectly imitable.

4. *Substitutes* in that the resource cannot have strategically equivalent substitutes that are neither rare nor imperfectly imitable.

Barney considers these attributes to be indicators of how heterogeneous and immobile a firm’s resources are and thus can help determine how useful they are in generating a sustained competitive advantage (Barney, 1991). He summarizes the relationship between these attributes in figure 15:

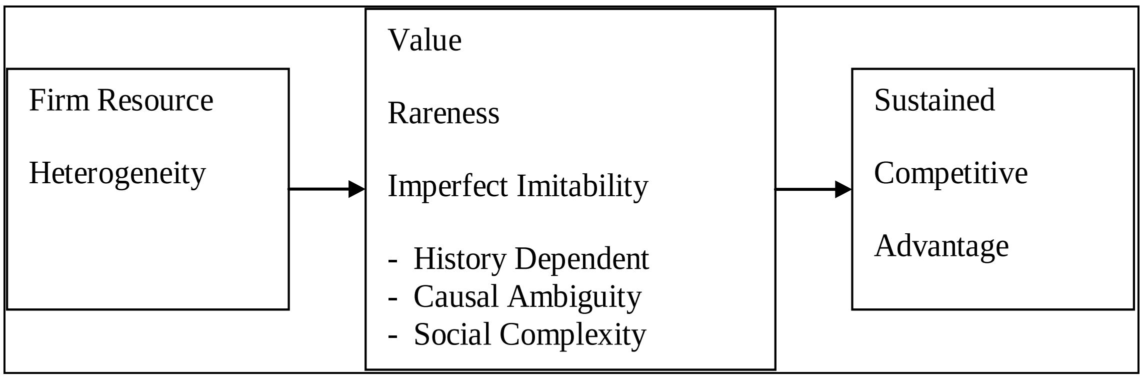

Figure 15. Resource-based View of the Firm Attributes (Barney, 1991).

It may thus be viewed that agile-based software development teams are firm resources that have the potential for generating sustained competitive advantage.

Summary

The literature review has culminated in a better understanding of the impact that IT has had in revolutionizing business productivity and innovation. However, the adoption of associated ‘business process change' by individuals has received little research attention as yet to identify the drivers of process adoption by employees, the factors influencing resistance, the impacts of process changes on employees, and potential interventions to ease transition (Venkatesh, 2006). Chapter III describes the research model, methodology, and variables used in this study.

Chapter III

Methodology

Introduction

Based on the theoretical and conceptual discussions in Chapter II, this chapter presents the proposed research model, shown in Figure 16, which is the basis for the contents of this chapter. Included in the chapter are the research rational, the research methodology, the research questions and hypotheses, the independent and dependent variables, the research instrument reliability and validity processes, and the method of analysis.

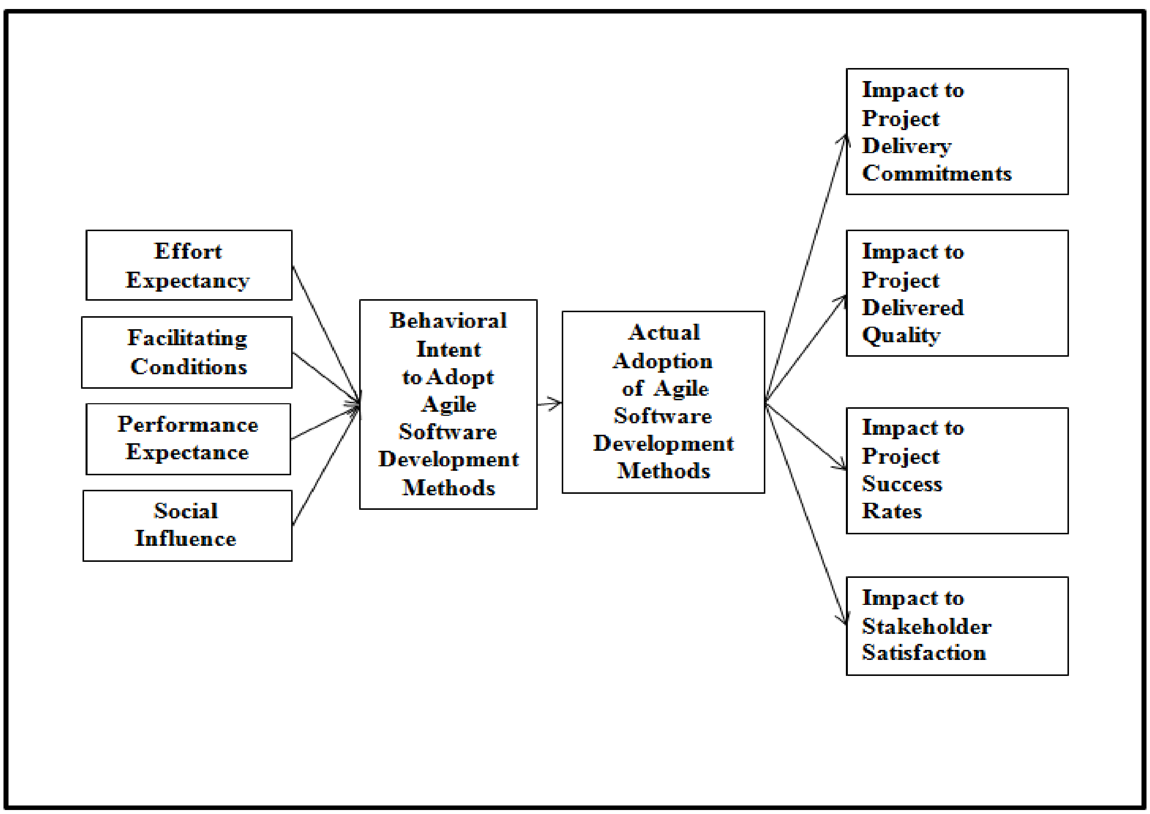

Figure 16. Proposed Research Model.

Research Rational

Agile software development methods, characterized by delivering customer value via incremental and iterative time-boxed development processes, have moved into the mainstream of the Information Technology (IT) industry (Cockburn, 2001). In addition, agile software development method researchers contend that software development is analogous to new product development and as such, is better served by novel and creative approaches which accommodate high rates of change, is not predicable, and requires adaptive methods to provide competitive advantages for organizations (Larman, 2004). However, despite a growing body of research which suggests that a predictive manufacturing approach, with big up-front specifications, detailed estimates, and speculative plans applicable to manufacturing domains, is the wrong paradigm for software development, many IT organizations are still hesitant to adopt agile approaches (Larman, 2004).

System development methodologies have continued to evolve as evidenced by research into method engineering (Siau, 1999). As a result, there is a plethora of system development methods available. However, the high failure rates of systems development efforts (Hirsch, 2002) and the complexity of many traditional system development methods, suggest that choosing the right method is a complex and difficult task. As a result, research suggests that a one-size-fits-all software development process does not fit the often turbulent nature of software development (Erickson & Siau, 2003; Erickson, Lyytinen, & Siau, 2005), and that agile methods would appear to be a better fit than traditional methods. However, adoption of agile software development methods remains in a lagging state and may be impacted by the following issues:

1. Resistance to agile software development methods has been primarily attributed to previous experience with traditional software development methods which posit that software intensive projects can be developed in a predictable style which uses a plan-driven, waterfall approach irrespective of the high failure rates of this approach when applied to software development projects (Beck, 1999; Boehm, 2002; Schwaber, 2001; Cockburn, 2002).
2. There is a paucity of research related to the study of agile software development methods, relative to their adoption and use, to provide additional insight into the nature of agile method adoption by organizations as well as research to help organizations better understand the impact of adopting these methods.

Venkatesh, Davis, and Morris (2007) have posited that although adoption models were originally developed to study technology, they have been extended far beyond their original boundaries to study adoption in such domains as dairy farming and green electricity use. Thus, they put forth the notion that the next step for researchers may be to develop a framework driven set of future research directions that can leverage current knowledge for solving today's relevant business problems. Venkatesh (2006) concluded that while individual-level technology adoption is undoubtedly mature, there are broader areas that can build upon technology adoption research by leveraging robust models where available.

Understanding the factors that influence the adoption of a software development method innovation, and using those factors to influence the adoption, should help to make implementation and use more successful. A proposed research design would include using the UTAUT as the basis for extending research to determine if relationships exist between factors of the UTAUT and perceived characteristic related to the adoption and use of agile software development methods.

Research Methodology

The research methodology, based on the proposed research model in Figure 16, uses the factors of the UTAUT (performance expectancy, effort expectancy, social influence, and facilitating conditions) as the basis for extending the UTAUT into the domain of business processes. Specifically, to investigate relationships between the UTAUT factors and the perceived characteristics related to ‘behavioral intent’ for adopting agile software development methods, where behavioral intent is a measure of the likelihood that a person will adopt an innovation where intentions do predict actions (Ajzen & Fishbein, 1980). Additional research was included which investigated the relationship between agile software development method adoption and the impact of adoption on key project performance attributes (on-time delivery of project functionality, post-delivery defects, project stakeholder satisfaction, and project success rates). As a quantitative study, it is based on testing a theory using statistical procedures to determine if generalizations of prediction can be developed which describe the validity of a theory (Creswell, 1994).

The research methodology encompassed the following steps that contributed to the development of the research questions and hypotheses:

1. *Preliminary Literature Review* which consisted of reviewing relevant research literature to identify where opportunities for research extension existed.
2. *Development of Research Questions* to qualify the topic of relevant research.

3. *Literature Review* which consisted of an in-depth review of research relevant to the research topic and questions.
4. *Development of Pilot Instrument* which consisted of extending the UTAUT scales to account for the relevant characteristics of the business processes associated with the intended adoption and use of agile software development methods as well as results from adoption.
5. *Survey Pilot and Instrument and Validation* which consisted of validating the reliability and validity of the survey questions via pre-test, modification, and finalization of the survey instrument.
6. *Field Survey* which consisted of a web-based survey consisting of 65 questions used for data collection purposes.
7. *Analysis* which consisted of analyzing the survey data via statistical techniques to test the research hypotheses.
8. *Conclusions* which consisted of a summarization of the research results, study limitations, and identification of areas for future research.

Research Questions and Hypotheses

The research questions are based on a preliminary and in-depth review of technology adoption and software development method adoption research. This review culminated in two primary research streams; extending the UTAUT factors into the domain of agile software development method adoption to determine their relationship on behavioral intent to adopt these methods, and investigating the impact of agile software development method adoption on project performance attributes. The specific research questions are as follows:

1. Is there a correlation between performance expectancy and the behavioral intent to adopt agile software development methods?
2. Is there a correlation between effort expectancy and the behavioral intent to adopt agile software development methods?
3. Is there a correlation between social influence and the behavioral intent to adopt agile software development methods?
4. Is there a correlation between facilitating conditions and the behavioral intent to adopt agile software development methods?
5. Is there a correlation between agile software development method adoption and project performance attributes of on-time delivery of project functionality, post-delivery defects (quality), stakeholder satisfaction (project team and customer), and project success rates (delivered versus cancelled projects)?

The specific research hypotheses are as follows:

Performance Expectancy

Hypothesis 1

- H_{1o}: Performance expectancy is negatively or not correlated to the adoption of agile software development methods.
- H_{1a}: Performance expectancy is positively correlated to the adoption of agile software development methods.

Effort Expectancy

Hypothesis 2

- H_{2o}: Effort expectancy is negatively or not correlated to the adoption of agile software development methods.
- H_{2a}: Effort expectancy is positively correlated to the adoption of agile software development methods.

Social Influence

Hypothesis 3

- H_{3o}: Social influence is negatively or not correlated to the adoption of agile software development methods.
- H_{3a}: Social influence is positively correlated to the adoption of agile software development methods.

Facilitating Conditions

Hypothesis 4

- H_{4o}: Facilitating conditions are negatively or not correlated to the adoption of agile software development methods.
- H_{4a}: Facilitating conditions are positively correlated to the adoption of agile software development methods.

Project Performance

Hypothesis 5

- H_{5o}: The use of agile software development methods are negatively or not correlated to increases in on-time delivery of project functionality.
- H_{5a}: The use of agile software development methods are positively correlated to increases in on-time delivery of project functionality.

Hypothesis 6

- H_{6o}: The use of agile software development methods are negatively or not correlated to decreases in project post-delivery defects.
- H_{6a}: The use of agile software development methods are positively correlated to decreases in project post-delivery defects.

Hypothesis 7

- H_{7o}: The use of agile software development methods are negatively or not correlated to increases in project stakeholder (project team and customer) satisfaction levels.
- H_{7a}: The use of agile software development methods are positively correlated to increases in project stakeholder (project team and customer) satisfaction levels.

Hypothesis 8

- H_{8o}: The use of agile software development methods are negatively or not correlated to improved success of projects (delivered versus cancelled projects).
- H_{8a}: The use of agile software development methods are positively correlated to improved success of projects (delivered versus cancelled projects).

The extension of the UTAUT factors for this study required modification to the original UTAUT survey questions. This was accomplished by replacing the terms 'the system' in the original UTAUT survey with the term 'agile software development methods' for this study.

Independent and Dependent Variables

The UTAUT integrated previous acceptance models and posits four direct determinants of usage intention and behavior and up to four moderators of key relationships (gender, age, experience, voluntariness of use). In this study, the moderators of key relationships were not analyzed; however data for these relationships was collected for future research.

The independent and dependent variables, operationalized for describing the UTAUT factors which may impact adoption of agile software development method as well as the results from agile software development method adoption, are identified in Table 1.

Table 1

Survey Variables

Independent Variables	
Variable Name	Description
Performance Expectancy	The degree to which an individual believes that using agile software development methods will help him or her to attain gains in job performance.
Effort Expectancy	The degree of ease associated with the ease of using agile software development methods.
Social Influence	The degree to which an individual perceives that important others believe he or she should use agile software development methods.
Facilitating Conditions	The degree to which an individual believes that an organizational and technical infrastructure exists to support the use of agile software development

Table 1

Survey Variables

Agile Software Development Method Adoption	methods. Number of months using agile software development methods.
Dependent Variables	
Variable Name	Description
Behavioral Intent	The behavioral intent to adopt agile software development methods.
On-time Delivery of Functionality	On-time delivery of project functionality resulting from agile software development method adoption.
Post-delivery Defects	Post-delivery defects resulting from agile software development method adoption.
Stakeholder Satisfaction	Project stakeholder satisfaction project team and customer satisfaction levels resulting from agile software development method adoption.
Project Success Rates	Project success rates delivered versus cancelled projects resulting from agile software development method adoption.

Table 1. Survey Variables.

Research Instrument

The variables that were operationalized for this study were pre-tested using a group of IT colleagues with various IT background ranging from utilities, financial services, education, and IT related consulting and training. They were experienced in a range of traditional and agile software development methods and as such, were able to provide substantive feedback on the initial questions. Their feedback was subsequently incorporated into the survey in its final formulation (Appendix A Research Instrument) which contains the survey questions. The sampling frame consisted of the following constitutes who have used, or may be imminently planning to use, agile software development methods:

1. Members of the global Software Process Improvement Network (SPIN) chapters as identified by the Software Engineering Institute at Carnegie Melon University (http://www.sei.cmu.edu/collaborating/spins/).
2. Members of the global Agile Methodology User Groups identified as by the Agile Alliance (http://www.agilealliance.org/show/1641)
3. Various I.T. industry conference promoters and presenters as identified by the Information Technology Worldwide Conferences website (http://www.conferencealerts.com/it.htm)

This sampling frame ensured un-biased results in the survey as these constitutes are familiar with IT related terminology, methods, and technologies which qualify them as an "informed audience" and as such, should improve the validity of the research (Fink, 2000). The survey cover letter (Appendix B), containing the rational for the survey and the link to the online location of the survey, was sent to the chairperson of the SPIN, Agile User Groups, and promoters and presenters at the various IT conferences via email to solicit participation in the survey.

An e-mail distribution is not generally appropriate for surveys where the total population is unlikely to have email access. However, in this research, the SPIN chapter members, Agile User Group members, and promoters and presenters at the various IT conferences are all likely to have email (Schaefer & Dillman, 1998), so distribution of the survey purpose and on-line location via an email was more cost effective and faster than using traditional survey methods of telephone and postal mail (Schaefer & Dillman; Schleyer & Forrest, 2000).

Members of the SPIN chapters, Agile User Groups, and promoters and presenters at the various IT conferences constitute the population for this research. The confidence level for this study was 95%; the level of significance was five%. The population is estimated to be approximately 1800 members. Given these values, the statistically significant random sample size was 278.

The factors utilized in this study were measured via a 65-question multi-item web-based adaptive survey using the UTAUT scale (Venkatesh, Morris, Davis, & Davis, 2003) as the basis for developing the questions pertinent to this research. An adaptive survey (Sheehan & Hoy, 1999) utilizes a series of questions to respondents based on answers to previous questions and allows participants to skip questions that are not relevant to them (Schonlau, Fricker, & Elliot, 2000). Due diligence was taken in the layout and organization of the survey to aid readability and usability which can impact the survey return rate (Morrel-Samuels, 2002). The survey was scripted to ensure that answers to questions are provided in the appropriate format to minimize the likelihood of inaccurate data. This was accomplished by on-line error validation provided by the web-based survey hosting tool (www.zoomerang.com). All questions were validated to ensure that there was no missing or incomplete data. If a survey participant chose not to complete the survey then their survey record was not used in the analysis of results.

The survey contained randomized questions adopted from previous scales developed by Davis (1989), and Venkatesh, Morris, Davis, and Davis (2003), and modified to fit this research by modifying the wording to reflect references to agile software development methods. The survey contains four sections pertaining to the research topic of interest as follows:

1. Section A: Current Users of Agile Software Development Methods
    - Performance Expectancy – four questions
    - Effort Expectancy – four questions
    - Social Influence – four questions
    - Facilitating Conditions – three questions
    - Project Impact – five questions
    - Management Support, Development Practices, and Project Team Characteristics - twelve questions
2. Section B: Future Adoption
    - Performance Expectancy – three questions
    - Effort Expectancy – four questions
    - Social Influence – four questions
    - Facilitating Conditions – three questions
    - Project Impact – five questions

3. Section C: Organizational Characteristics

    - Organizational Cultural Type - one question
    - Organizational Learning Orientation - one question
    - Improving Software Development Processes – one question

4. Section D: Demographic

    - Geographic Work Region – one question
    - Industry Classification – one question
    - IT Organization Size – one question
    - Role in Organization – one question
    - Years of Software Development Experience – one question
    - Age – one question
    - Level of Education – one question
    - Gender – one question

The instrument items were coded via a two-part identification number reflecting the survey section (part one), and its position in the survey section (part two). For example, question "A-4" reflects question 4 in section A of the survey. The flow of the survey questions is illustrated in the Figure 17.

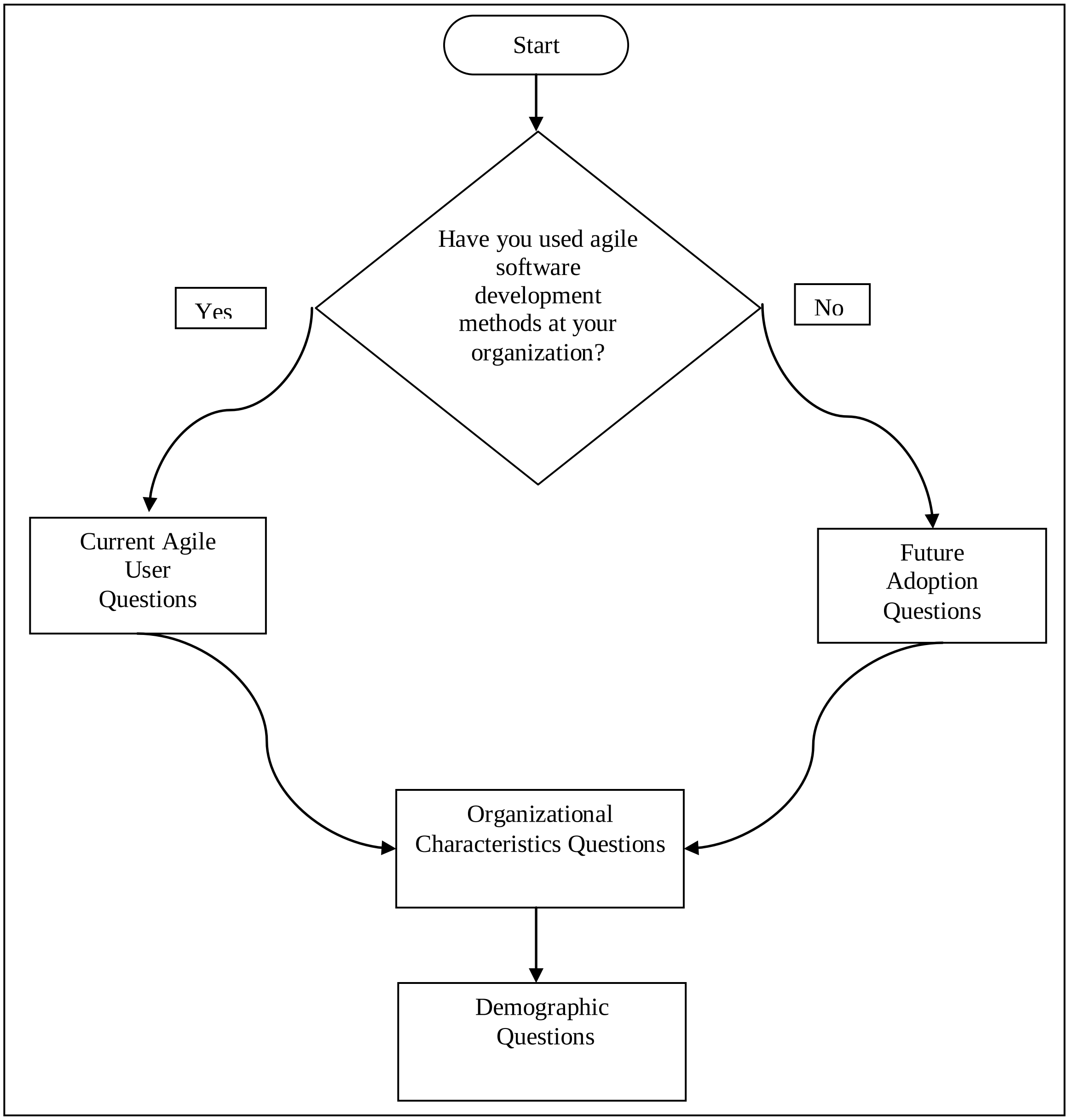

Figure 17. Flow Diagram of Survey Questions.

Feedback was provided at the end of each section of the survey indicating to participants their percent of progress in completing the survey. Researchers (Crawford, Couper, & Lamis, 2001) found that feedback improved survey completion rates by 3.5 %. Demographic questions were placed at the end of the survey to also improve response rate (Punter, Ciolkowski, Freimut, & John, 2003).

Instrument Reliability and Validity

The instrument was based on previously validated scales developed by Davis (1989), and Venkatesh, Morris, Davis, and Davis (2003), as well as new scale items to reflect the impact of agile method adoption on select characteristic of project performance. The instrument utilized a 5-point Likert scale as follows:

1 = strongly disagree

2 = disagree

3 = undecided

4 = agree

5 = strongly agree

Utilizing a 5-point Likert scale produces higher reliability scores than 3- or 7- point Likert scales (Dyba, 2000). A survey readability panel, consisting of professional colleagues and agile methodology experts, was used to pre-test the instrument for content validity (Dyba, 2000). Feedback was incorporated into the instrument. The instrument was also tested for reliability and internal consistency via Cronbach's Alpha (Cronbach, 1951). Construct validity was also tested by performing factor analysis via Varimax rotation.

Method of Analysis

The method of analysis used in this study was a descriptive correlation and regression analysis to statistically test the research hypotheses. The data collected was organized and coded via Microsoft Excel and then imported into PASW Statistics GradPack 18 software (formally SPSS) for analysis and presentation via text, tables, and graphical formats to convey the results of the analysis.

Correlational studies enable researchers to determine if relationships exist between two or more variables, and to determine the direction (positive or negative) and strength of the relationships (Sekaran, 2000). The most common correlation for use with two ordinal variables or an ordinal and an interval variable is Spearman's rho (Fink, 2003). Rho for ranked data equals Pearson's r for interval data (Norusis, 2009). The formula for Spearman's rho is: $rho = 1 - [(6*SUM(d^2)/n(n^2 - 1)]$, where d is the difference in ranks.

A statistical model was generated to test the hypotheses examining the relationship between the independent variables of performance expectancy, effort expectancy, social influence, facilitating conditions, and agile adoption, to the dependant variables of behavioral intent to adopt agile software development methods and project performance (on-time delivery of project functionality, post-delivery defects, project stakeholder satisfaction, and project success rates).

Chapter IV

Analysis and Presentation of Findings

Introduction

Chapter III describes the research methodology that was employed for this study. This chapter presents the results from the research methodology in the following sections:

1. *Survey size, data collection approach, and survey demographics* which presents the survey sample size and response rate, data collection and data coding process, and demographics in terms of participation by work region, industry, role, software development experience, age, and gender.
2. *Instrument validity and reliability, and descriptive statistics* which presents the results of the instrument validation and reliability test and descriptive statistics.
3. *Analytic techniques and findings* which presents the analysis methods employed and results of the hypotheses testing.
4. *Summary* which present the results of this study as they relate to the research question developed.

Survey Size, Data Collection Approach, and Survey Demographics

In this study, a total of 333 surveys were obtained using a web-based adaptive survey which utilized a series of questions to collect data. An adaptive survey allows participants to skip questions that are not relevant to them (Schonlau, Fricker, & Elliot, 2000).

Surveys with missing data can cause havoc in statistical analysis. As a result, from the total of 333 surveys, 161 were validated as complete surveys and were utilized in the statistical analysis and findings. This constitutes a 48.3% completion rate for the total surveys attempted in the sample.

The survey data in this study was developed by the researcher using an Internet-based commercial survey tool (www.zoomerang.com) and was made available for online completion via the Internet from September 1st, 2010 to November 15th, 2010. The data from the surveys was subsequently exported into Microsoft Excel for initial validation and variable coding. The coded data was then imported into PASW Statistics GradPack 18 software (formally SPSS) for analysis and presentation.

In this study, the survey questions initially developed to test factors posited in the Unified Theory of Acceptance and Use of Technology (Venkatesh, Morris, Davis, & Davis, 2003) were modified to investigate the relationship between the UTAUT factors and the perceived characteristics related to ‘behavioral intent’ for adopting agile software development methods, where behavioral intent is a measure of the likelihood that a person will adopt an innovation where intentions do predict actions (Ajzen & Fishbein, 1980). The responses for these questions were rated using a 5-point Likert scale, ranging from ‘strongly disagree’ code as 1, and increasing in range to ‘strongly agree’ coded as 5. The use of a 5-point Likert scale produces higher reliability scores than 3 or 7 point Likert scales (Dyba, 2000).

Demographic and organizational information also was collected from the survey participants across eight categories and provides useful insight into the backgrounds of the participants. This information was not analyzed as part of the hypotheses testing, but is intended for use in future research.

Table 2 presents demographics of survey participants by geographic work region.

Table 2
Geographic Work Regions

		Frequency	Percent
Valid	Africa	9	5.6
	Asia	3	1.9
	Eastern Europe	2	1.2
	European Union	49	30.4
	Middle East	1	0.6
	North America	86	53.4
	Oceania	1	0.6
	South America	10	6.2
	Total	161	100.0

Table 2. Geographic Work Regions.

The top three regions are North America (53.4%), European Union (30.4%), and South America (6.2 %).

Table 3 presents demographics of survey participants by industry classification.

Table 3
Industry Classifications

		Frequency	Percent
Valid	Agriculture	1	0.6
	Avionics	1	0.6
	Banking	44	27.3
	Consulting	14	8.7
	Consumer electronics	1	0.6
	EDA Electronic Design Automation	1	0.6
	Education	3	1.9
	Engineering, EDA, CAD	1	0.6
	Financial Services (other than banking)	16	9.9
	Gaming	1	0.6
	Government	6	3.7

	Health Care	7	4.3
	Information Technology	37	23.0
	Manufacturing	6	3.7
	Market research	1	0.6
	Media	1	0.6
	Oil & Gas	1	0.6
	Pharmaceutical Drug Development	1	0.6
	Research	1	0.6
	Retail	3	1.9
	semiconductors and software	1	0.6
	Telecommunications/Internet Service Provider	7	4.3
	Transportation	1	0.6
	Utility	5	3.1
	Total	161	100.0

Table 3. Industry Classifications.

The top five classifications are Banking (27.3%), Information Technology (23%), Financial Services (9.9%), Consulting (8.7%), and Telecommunications/Internet Service Providers (4.3%). Collectively, these classifications reflect a wide variety of industries represented.

Table 4 presents demographics of survey participants by organizational role.

Table 4
Organizational Role

		Frequency	Percent
Valid	Agile Coach	12	7.5
	Architect	13	8.1
	Business Analyst	3	1.9
	Business Partner/IT User	3	1.9
	Chief Information Officer (CIO)	1	0.6
	Chief Technology Officer (CTO)	1	0.6
	Client Manager	1	0.6
	Consultant	14	8.7
	Developer	21	13.0
	Development Manager	16	9.9
	Director	10	6.2

	Functional Manager	1	0.6
	IT Business System Analyst	1	0.6
	IT Staff	3	1.9
	Problem Manager	1	0.6
	Process Engineer	1	0.6
	Process Manager	1	0.6
	Program Manager	5	3.1
	Project Manager	27	16.8
	Quality and Process	1	0.6
	Quality Assurance	8	5.0
	Team Lead	15	9.3
	Tester	2	1.2
	Total	161	100.0

*Table 4.*Organizational Role.

The top five roles are Project Manager (16.8%), Developer (13%), Development Manager (9.9%), Team Lead (9.3%), and Architect (8.1%). Collectively, these roles reflect a wide variety of functions established the IT industry.

Table 5 presents demographics by ranges of years of software development experience.

Table 5
Years of Software Development Experience

		Frequency	Percent
Valid	1 - 2	4	2.5
	3 - 5	20	12.4
	6 - 10	24	14.9
	11 - 15	42	26.1
	> 15	61	37.9
	Not applicable	10	6.2
	Total	161	100.0

*Table 5.*Years of Software Development Experience.

The top three ranges are > 15 years (37.9), 11 – 15 years (26.1%), and 6-10 years (14.9%). Over 60% of the survey participants have 11 or more years of software development experience. Collectively, the age ranges reflect considerable depth in the number of years of software development experience accumulated by the survey participants.

Table 6 presents demographics of survey participants by ranges of physical age. For example, 2 indicates an age range of less than 30 years, 3 indicates an age range of 30 – 39 years and so on to the upper range of 60 or more years in physical age.

Table 6

Age Ranges

		Frequency	Percent
Valid	2	20	12.4
	3	52	32.3
	4	51	31.7
	5	30	18.6
	6	8	5.0
	Total	161	100.0

Table 6. Age Ranges.

The top three ranges are 30 – 39 years (32.3%), 40 – 49 years (31.7%), and 50 – 59 years (18.6%). Collectively, these age ranges reflect considerable physical maturity in the survey participants.

Table 7 presents demographics of survey participants by education levels.

Table 7

Education Levels

		Frequency	Percent
Valid	High school or General Equivalency Diploma (GED)	8	5.0
	2-year college degree	10	6.2
	4-year college degree	74	46.0
	Masters degree	62	38.5
	Doctoral degree	7	4.3
	Total	161	100.0

Table 7. Education Levels.

The top three levels are 4-year college degree (46%), master degree (38.5%), and 2-year college degree (6.2%). Collectively, these levels reflect that the vast majority of survey participants (84.5%) have 4 or more years of college.

Table 8 presents demographics of survey participants by gender.

Table 8
Gender

		Frequency	Percent
Valid	Female	30	18.6
	Male	131	81.4
	Total	161	100.0

Table 8. Gender.

The majority of the survey participants are male (81.4 %) with females (18.6%) accounting for the minority. These demographics are similar to those found in the 2009 Agile Practitioner Salary Survey (VersionOne, 2009) where 13% were female and 87% were male.

Instrument Validity, Reliability, and Descriptive Statistics

Validity, in conventional terms, refers to the extent to which an empirical measure adequately reflects the real meaning of the concept under consideration (Babbie, 2004). In this study, construct validity of the survey was tested by performing factor analysis (via Varimax rotation). In the factor analysis performed, the Kaiser-Meyer-Olkin Measure of Sampling Adequacy (MSA), which measures the amount of variance in a variable that is accounted for by the other variables (Norusis, 2009), scored above 0.5 overall for the variables analyzed and the factor loading confirmed dimensionality based on the total variance explained.

Reliability, in the abstract, is a matter of whether a particular technique, applied repeatable to the same object, yields the same result each time (Babbie, 2004). In survey development, the most widely used measure of internal consistency is the Cronbach's Alpha reliability coefficient (Cronbach, 1951), where the coefficient value of 0.70 is generally considered to be the lowest acceptable limit for exploratory research. In this study, variables were operationalized based on the factors of the UTAUT to reflect the 'behavioral intent' to adopt agile software development methods. Additional variables were operationalized to measure the factor of 'project performance' based on actual adoption of agile software development methods.

The variables developed consisted of survey items that were summated and tested for internal consistency (Norusis, 2009). In this study, the factors of the UTAUT were extended into the realm of business processes as posited by Venkatesh, Davis and Morris (2007) when they described next steps for UTAUT related research in term of 'developing a framework driven set of future research directions to leverage current knowledge for solving today's relevant business problems.'

Table 9 presents the Cronbach's Alpha reliability coefficients for this study.

Table 9

Cronbach's Alpha Reliability Coefficients

Variables	Number of items	Reliability Coefficient
Performance Expectancy	11	.85
Effort Expectancy	11	.80
Social Influence	12	.81
Facilitating Conditions	12	.83
Behavioral intent	10	.78
Project Performance	10	.83

Table 9. Cronbach's Alphas Reliability Coefficients.

The Cronbach's Alphas reliability coefficients for all variables met the generally acceptable minimum level of 0.70.

Table 10 presents the descriptive statistics from this study.

Table 10
Descriptive Statistics

Variables	N	Minimum	Maximum	Mean	Std. Deviation
Performance Expectancy	161	1.50	5.00	3.69	.63
Effort Expectancy	161	2.25	5.00	3.70	.54
Social Influence	161	1.00	5.00	3.51	.80
Facilitating Conditions	161	1.50	5.00	3.54	.60
Behavioral Intent	161	1.00	5.00	3.72	.54
On-Time Delivery of Project Functionality	161	1.00	5.00	3.93	.88
Post-delivery Defects	161	1.00	5.00	3.62	1.01
Stakeholder Satisfaction	161	1.00	5.00	3.83	.72
Project Success Rates	161	1.00	5.00	3.73	.91

*Table 10.*Descriptive Statistics.

Overall, the mean scores for the factors are positive with regard to behavioral intent to adopt agile software development methods as well as adoption impact on project performance as operationalized in this study. On-time delivery of project functionality has the highest mean, or most agreeable value, with a value of 3.93 and a standard deviation of .88. Social Influence has the lowest mean at 3.51 with a standard deviation of .80.

Analytic Techniques

Correlation analysis enables researchers to determine if relationships exist between two or more variables, and to determine the direction (positive or negative) and strength of the relationships (Sekaran, 2000). The most common correlation measure for ordinal variables is Spearman's rho (Fink, 2003). Rho for ranked data equals Pearson's r for numeric data (Norusis, 2009).

The formula for Spearman's rho is: $rho = 1 - [(6*SUM(d^2)/n(n^2 - 1)]$, where d is the difference in ranks. In this study, Spearman's rho correlation coefficients were calculated to assess the internal validity of the factors and to determine the strength and direction of the relationships between the independent variables of performance expectancy, effort expectancy, social influence, and facilitation conditions that were measured against the dependent variable of behavioral intent (to adopt agile software development methods). Table 11 presents the bivariate Spearman's rho correlations coefficients and the significance level for these variables.

Table 11
Spearman's rho Correlations – Behavioral Intent

Variables		Performance Expectancy	Effort Expectancy	Social Influence	Facilitating Conditions	Behavioral Intent
Performance Expectancy	Correlation Coefficient	1.000	.345	.342	.413	**.667**
	Sig. (1-tailed)	.	.000	.000	.000	.000
Effort Expectancy	Correlation Coefficient	.345	1.000	.322	.459	**.583**
	Sig. (1-tailed)	.000	.	.000	.000	.000
Social Influence	Correlation Coefficient	.342	.322	1.000	.684	**.671**
	Sig. (1-tailed)	.000	.000	.	.000	.000
Facilitating Conditions	Correlation Coefficient	.413	.459	.684	1.000	**.579**
	Sig. (1-tailed)	.000	.000	.000	.	.000
Behavioral Intent	Correlation Coefficient	.667	.583	.671	.579	1.000
	Sig. (1-tailed)	.000	.000	.000	.000	.

Table 11. Spearman's rho Correlations – Behavioral Intent.

The analysis of the correlation coefficients in Table 11 reflects the strength and direction of the relationships between the independent variables and dependant variable as follows:

1. Performance expectancy has a strong positive correlation to behavioral intent.
2. Effort expectancy has a positive correlation to behavioral intent.
3. Social influence has a strong positive correlation to behavioral intent.
4. Facilitating conditions has a positive correlation to behavioral intent.

This study also included research to investigate the relationship between agile software development method adoption (number of months using agile software development methods) and adoption impact on project performance attributes (on-time delivery of project functionality, post-delivery defects, project stakeholder satisfaction, and project success rates). Table 12 presents the bivariate Spearman's rho correlations coefficients and the significance level for these variables.

Table 12
Spearman's rho Correlations – Agile Adoption

Variables		Number of months using agile software development methods	On-time Delivery	Post Deliver Defects	Stakeholder Satisfaction	Project Success Rates
Number of months using agile software development methods	Correlation Coefficient	1	**.166**	**.066**	**.153**	**.058**
	Sig. (1-tailed)	.	.030	.228	.042	.257
On-time Delivery	Correlation Coefficient	.166	1	.545	.685	.556
	Sig. (1-tailed)	.030	.	.000	.000	.000

Post Deliver Defects	Correlation Coefficient	.066	.545	1	.539	.548
	Sig. (1-tailed)	.228	.000	.	.000	.000
Stakeholder Satisfaction	Correlation Coefficient	.153	.685	.539	1	.625
	Sig. (1-tailed)	.042	.000	.000	.	.000
Project Success Rates	Correlation Coefficient	.058	.556	.548	.625	1
	Sig. (1-tailed)	.257	.000	.000	.000	.

Table 12. Spearman's rho Correlations – Agile Adoption.

The analysis of the correlation coefficients in Table 12 reflects the strength and direction of the relationships between the independent variables and the dependant variables as follows:

1. Number of months using agile software development methods had a positive correlation to on-time delivery of project functionality.
2. Number of months using agile software development methods had a weak positive correlation to post-delivery defects.
3. Number of months using agile software development methods had a positive correlation to project stakeholder satisfaction.
4. Number of months using agile software development methods had a weak positive correlation to project success rates.

Analysis of Hypotheses

The hypotheses in this study were designed to test whether a variable is positively or negatively correlated with another. This can be demonstrated by observing the correlations coefficients of the variables to determine the magnitude of the relationship as well as the strength of the relationships (Brightman & Schneider, 1994).

The entry labeled *Sig.* (1-tailed) in Tables 12 and 13 is the observed significance level for the test of the hypotheses that the population value for the correlation coefficient is zero (0), and coefficients that have observed significance levels smaller than 0.05 are determined to be significant (Norusis, 2009).

Using the observed significance levels presented in Tables 12 and 13, we can now test hypotheses associated with the following research questions.

1. Is there a correlation between performance expectancy and the behavioral intent to adopt agile software development methods?

   Hypothesis 1

   - H_{1o}: Performance expectancy is negatively or not correlated to the adoption of agile software development methods.
   - H_{1a}: Performance expectancy is positively correlated to the adoption of agile software development methods.

2. Is there a correlation between effort expectancy and the behavioral intent to adopt agile software development methods?

   Hypothesis 2

   - H_{2o}: Effort expectancy is negatively or not correlated to the adoption of agile software development methods.
   - H_{2a}: Effort expectancy is positively correlated to the adoption of agile software development methods.

3. Is there a correlation between social influence and the behavioral intent to adopt agile software development methods?

Hypothesis 3

- H_{3o}: Social influence is negatively or not correlated to the adoption of agile software development methods.
- H_{3a}: Social influence is positively correlated to the adoption of agile software development methods.

4. Is there a correlation between facilitating conditions and the behavioral intent to adopt agile software development methods?

Hypothesis 4

- H_{4o}: Facilitating conditions are negatively or not correlated to the adoption of agile software development methods.
- H_{4a}: Facilitating conditions are positively correlated to the adoption of agile software development methods.

5. Is there a correlation between agile software development method adoption and project performance attributes of on-time delivery of project functionality, post-delivery defects (quality), stakeholder satisfaction (project team and customer), and project success rates (delivered versus cancelled projects)?

   Hypothesis 5

   - H_{5o}: The use of agile software development methods are negatively or not correlated to increases in on-time delivery of project functionality.
   - H_{5a}: The use of agile software development methods are positively correlated to increases in on-time delivery of project functionality.

Hypothesis 6

- H_{6o}: The use of agile software development methods are negatively or not correlated to decreases in project post-delivery defects.
- H_{6a}: The use of agile software development methods are positively correlated to decreases in project post-delivery defects.

Hypothesis 7

- H_{7o}: The use of agile software development methods are negatively or not correlated to increases in project stakeholder (project team and customer) satisfaction levels.
- H_{7a}: The use of agile software development methods are positively correlated to increases in project stakeholder (project team and customer) satisfaction levels.

Hypothesis 8

- H_{8o}: The use of agile software development methods are negatively or not correlated to improved success of projects (delivered versus cancelled projects).
- H_{8a}: The use of agile software development methods are positively correlated to improved success of projects (delivered versus cancelled projects).

Table 13 presents the observed correlation coefficient (r), the significant level of the test, the number corresponding to the hypotheses tested, and whether the null hypotheses should be accepted or rejected.

Table 13
Hypotheses Testing Results

Variables	Correlation Coefficient r	Significance Level (1-tailed) Alpha = .05	Hypotheses Tested	Accept or Reject Null
Performance Expectancy	.677	.000	1	Reject
Effort Expectancy	.583	.000	2	Reject
Social Influence	.671	.000	3	Reject
Facilitating Conditions	.579	.000	4	Reject
On-time delivery of project functionality	.166	.030	5	Reject
Post-delivery defects	.066	.228	6	Accept
Stakeholder satisfaction	.153	.042	7	Reject
Project success rates	.058	.257	8	Accept

Table 13. Hypotheses Testing Results.

As summarized in Table 13, the null hypotheses can be rejected for hypotheses one, two, three, four, and five, accepted for hypothesis six, rejected for hypothesis seven, and accepted for hypothesis eight. Therefore, according to this study performance expectancy, effort expectancy, social influence, and facilitating conditions are positively related to behavioral intent to adopt agile software development methods. The relationship of agile software development method adoption to project performance is mixed. According to this study, agile software development method adoption is positively related to on-time delivery of project functionality and stakeholder satisfaction, and not positively related to post delivery defects and project success rates.

Linear regression can be used to predict the value of dependent variables from the value of independent variables (Norusis, 2009). The linear regression model assumes that there is a linear, or "straight line," relationship between the dependent variable and each predictor or independent variable. This relationship is described in the following formula: $y_i=b_0+b_1x_{i1}+...+b_px_{ip}+e_i$

where

y_i is the value of the ith case of the dependent scale variable
p is the number of predictors
b_j is the value of the jth coefficient, $j=0,...,p$
x_{ij} is the value of the ith case of the jth predictor
e_i is the error in the observed value for the ith case

The model is linear because increasing the value of the jth predictor by 1 unit increases the value of the dependent by b_j units. Note that b_0 is the intercept, the model-predicted value of the dependent variable when the value of every predictor is equal to 0.

As presented in Table 13, there was a positive relationship between agile software development method adoption to on-time delivery of project functionality as well as a positive relationship between agile software development method adoption and stakeholder satisfaction. As a result, bivariate linear regression analysis was performed to determine if predictive models could be developed for the independent variable of agile software development method adoption and on-time delivery of project functionality, as well as agile software development method adoption and stakeholder satisfaction. Table 14 present the results of the bivariate linear regression analysis for the independent variable of agile software development method adoption and on-time delivery of project functionality.

Table 14

Regression Results – On-time Delivery of Project Functionality

Variables	Unstandardized Coefficients		Standardized Coefficients	Sig. Alpha = .05	R Square
	B	Std. Error	Beta		
(Constant)	3.428	.269		.000	
Number of months using agile software development methods	.144	.076	.166	.060	.028

Table 14. Regression Results – On-time Delivery of Project Functionality.

From Table 14, we can calculate a least-squares regression line for predicting the level of on-time delivery of project functionality based on number of month of agile software development experience as follows:

- On-time Delivery of Project Functionality = 3.428 + (.144 x number of month of agile software development experience).

While the model does provide predictive capability, the R square value of 3% is not adequately sufficient (≥ .70) to account for the proportion of variability in on-time delivery of project functionality based on the adoption of agile software development methods.

Table 15 present the results of the bivariate linear regression analysis for the independent variable of agile software development method adoption and stakeholder satisfaction.

Table 15
Regression Results – Stakeholder Satisfaction

Variables	Unstandardized Coefficients B	Unstandardized Coefficients Std. Error	Standardized Coefficients Beta	Sig. Alpha = .05	R Square
(Constant)	3.429	.220		.000	
Number of months using agile software development methods	.109	.062	.153	.084	. 023

Table 15. Regression Results – Stakeholder Satisfaction.

From Table 15, we can calculate a least-squares regression line for predicting the level of Stakeholder Satisfaction based on number of month of agile software development experience as follows:

- Stakeholder Satisfaction = 3.429 + (.109 x number of month of agile software development experience).

While the model does provide predictive capability, the R square value of 2% is not adequately sufficient ($\geq$ .70) to account for the proportion of variability in stakeholder satisfaction based on the adoption of agile software development methods.

Summary

In chapter IV, the survey size, data collection approach, and survey demographics were presented, along with the approaches used to determine the survey's validity and reliability. Also included were cross tabulations of demographic information on the survey participants. Descriptive statistical analysis of the variables was included and tests for zero correlation were performed for each of the hypotheses. Bivariate linear regression analysis was performed to determine if predictive models could be developed for the determining the expected level of on-time delivery of project functionality and stakeholder satisfaction based on the adoption of agile software development methods.

The test of the hypotheses that the population value for the correlation coefficient is zero led to the rejection of several hypotheses with the end result reflecting a mix of variables in the final model. Chapter V will include the final research model from the research as well as the summary, conclusions, and recommendations based on the information from Chapter IV.

Chapter V

Summary and Conclusions

Chapter Overview

While the use of agile software development methods has steadily increased in recent years, adoption remains in a lagging state and may be impacted by the following issues (Larman, 2004):

1. Resistance to agile software development methods has been primarily attributed to previous experience with traditional software development methods which posit that software intensive projects can be developed in a predictable style which uses a plan-driven, waterfall approach irrespective of the high failure rates of this approach when applied to software development projects (Beck, 1999; Boehm, 2002; Schwaber, 2001; Cockburn, 2002).
2. There is a paucity of research related to the study of agile software development methods, relative to their adoption and use, to provide additional insight into the nature of agile method adoption by organizations as well as research to help organizations better understand the impact of adopting these methods.

This study was conducted to identify factors that can help to mitigate concerns which have been identified by the Information Technology industry when considering the adoption of agile software development methods (Larman, 2004). Specifically, this study expands upon prior technology acceptance research to extend factors of the Unified

Theory of Acceptance and Use of Technology (UTAUT) (Venkatesh, Morris, Davis, & Davis, 2003) into the domain of business processes (Venkatesh, 2006) as they relate to the ‘behavioral intent’ to adopt agile software development methods, where behavioral intent is a measure of the likelihood that a person will adopt an innovation where intentions do predict actions (Ajzen & Fishbein, 1980). Additional research was undertaken to investigate the relationship between the adoption of agile software development methods and the impact of adoption on specific performance attributes of IT projects.

Significant Findings

The research questions for the study are as follows:

1. Is there a correlation between performance expectancy and the behavioral intent to adopt agile software development methods?
2. Is there a correlation between effort expectancy and the behavioral intent to adopt agile software development methods?
3. Is there a correlation between social influence and the behavioral intent to adopt agile software development methods?
4. Is there a correlation between facilitating conditions and the behavioral intent to adopt agile software development methods?
5. Is there a correlation between agile software development method adoption and project performance attributes of on-time delivery of project functionality, post-delivery defects (quality), stakeholder satisfaction (project team and customer), and project success rates (delivered versus cancelled projects)?

In the proposed research model identified in Figure 16, the factors of the UTAUT were hypothesized to have a positive correlation with behavioral intent to adopt agile software development methods. From the analysis performed in Chapter IV, it was found that the independent variables of performance expectancy, effort expectancy, social influence, and facilitating conditions were positively related to behavioral intent to adopt agile software development methods. This supports prior research where behavioral intent is a measure of the likelihood that a person will adopt an innovation where intentions do predict actions (Ajzen & Fishbein, 1980). This leads to the conclusion that organizations looking to adopt agile software development methods should address the factors of performance expectancy, effort expectancy, social influence, and facilitating conditions to positively influence adoption outcomes.

The proposed research model (Figure 16) also hypothesized that adoption of agile software development methods would positively impact key project performance attributes of on-time delivery of project functionality, post-delivery defects, project stakeholder satisfaction, and project success rates. From the analysis performed in Chapter IV, it was found that the independent variable of agile software development method adoption (number of months using agile software development methods) is positively related to project performance regarding on-time delivery of project functionality and stakeholder satisfaction, and not positively related to post delivery defects and project success rates. In addition, while predictive models were developed for determining expected positive results from adopting agile software development methods with regard to on-time delivery of project functionality and stakeholder satisfaction, the low values for the coefficient of determination (R squared) for these variables (3% and

2% respectively) were not adequately sufficient (≥ .70) to account for the proportion of variability in on-time delivery of project functionality or stakeholder satisfaction based on the adoption of agile software development methods. This leads to the conclusion that organizations looking to adopt agile software development methods should emphasize the positive impact of agile software development method adoption with regards to the on-time delivery of project functionality and stakeholder satisfaction levels.

Final Research Model

Based on the findings in the previous section, Figure 18 presents the final research model which incorporates the findings that were found in this study. The model also includes the correlation coefficients for the variables.

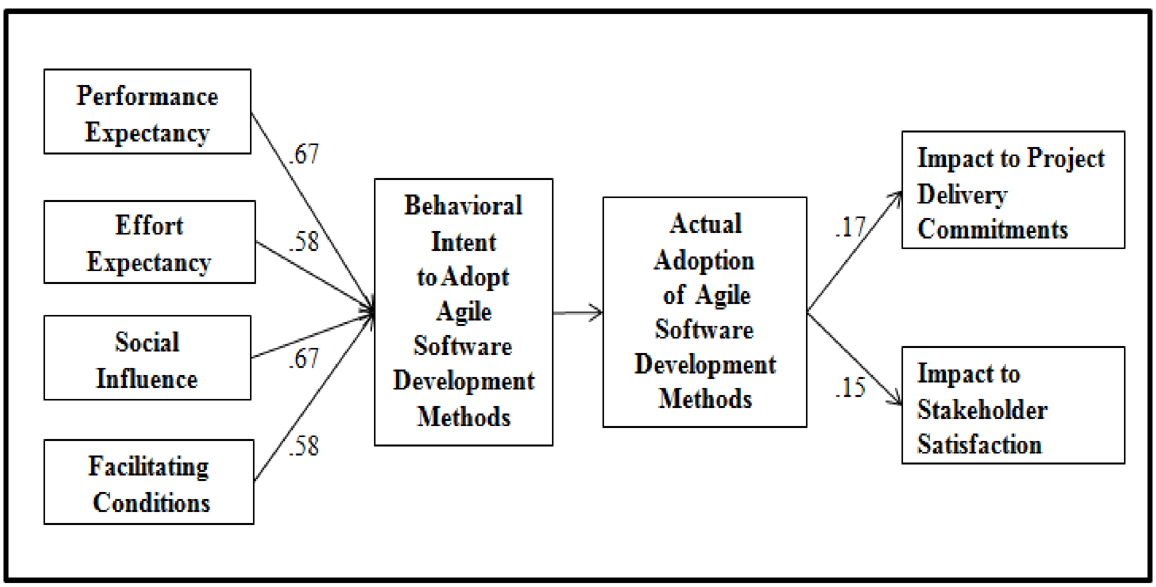

Figure 18. Final Research Model.

Implications From The Study

Based on the findings in this study, one can demonstrate that software development method adoption research has been enriched. In adding to this body of knowledge, this study demonstrates that organizations looking to pursue the adoption of agile software development methods should take into account the impact that performance expectancy, effort expectancy, social influence and facilitating conditions can have in influencing the behavioral intent of individuals to adopt agile software development methods. In addition, how the actual adoption of these methods can have a positive impact to on-time delivery of project functionality and stakeholder satisfaction.

The Resource Based Viewed (RBV) of the Firm (Barney, 1991) was developed to identify sources of sustained competitive advantage for firms based on resource advantages that may be available to a firm. As a result, one can surmise that IT organizations which understand the link between agile software development methods and sustained competitive advantage, should consider that adoption of these methods to gain this advantage.

Study Limitations

The limitations in this study are partially related to the research design which may also be source for future research. The specific limitations identified in this study are as follows:

1. This study did not analyze the four moderators of key relationships associated with the UTAUT factors (gender, age, experience, voluntariness of use) which may influence behavior intent to adopt agile software development methods.
2. This study did not analyze organizational characteristics such as industry type, cultural, and learning orientation which may influence behavior intent to adopt agile software development methods.

3. Some organizations have policies prohibiting participating in surveys of business processes, as they are seen as providing a competitive advantage.
4. This study was cross-sectional by design and as such, findings do not reflect the results of using agile software development methods on a year-over-year basis.

Future Research

Future avenues for research include the following areas:

1. Analyze the four moderators of key relationships associated with the UTAUT factors (gender, age, experience, voluntariness of use) to determine their impact on behavior intent to adopt agile software development methods.
2. Analyze demographic and organizational characteristics to determine their impact on behavior intent to adopt agile software development methods.
3. Analyze the demographic information collected to determine their influence on the results from adopting agile software development methods.
4. Research the results from using agile software development methods on a year-over-year basis.

Conclusions

The mission of profit oriented organizations should not be ‘making a profit’; the real mission should be ‘deciding what to do to make a profit.’ While agile software development methods may not provide IT organizations with an answer for deciding ‘what to do’, they may provide the answer for deciding ‘how to do it.’

APPENDIX

APPENDIX A

RESEARCH INSTRUMENT

Survey of Agile Software Development Method Adoption and Project Impact

Thank you for participating in a doctoral research project designed to extend the body of knowledge regarding factors influencing current and future adoption of agile software development methods, as well as the impact to key project attributes resulting from adoption.

The research involves completing a brief anonymous survey which should take no more than 10 minutes to complete. Participation in the survey is completely voluntary with no compensation or known risks.

To access the survey, select the following link:
Survey of Agile Software Development Method Adoption and Project Impact

Please submit inquires about this research to the following individuals via email:

Tracy Lambert (tlambert@nova.edu) - Nova Southeastern University

Dr. Richard McCarthy - (rmccarth@nova.edu) - Nova Southeastern University

Nova Southeastern University

AGILE SOFTWARE DEVELOPMENT METHOD ADOPTION AND IMPACT

Have you used agile software development methods at your organization?

If Yes – please select→ **Section A: Current users of Agile Software Development Methods.**

If No – please select → **Section B: Future Adoption.**

<u>Section A: Users of Agile Software Development Methods</u>

To the best of your ability, please select one answer to each of the following statements ***which may have contributed*** to the implementation of agile software development methods at your organization:

A-1 I find agile software development methods useful in my job.

☐ Strongly Disagree ☐ Disagree ☐ Undecided ☐ Agree ☐ Strongly Agree

A-2 I clearly understand how to use agile software development methods.

☐ Strongly Disagree ☐ Disagree ☐ Undecided ☐ Agree ☐ Strongly Agree

A-3 People who influence my behavior think that I should use agile software development methods.

☐ Strongly Disagree ☐ Disagree ☐ Undecided ☐ Agree ☐ Strongly Agree

A-4 I have the resources (support team, training, infrastructure) necessary to use agile software development methods.

☐ Strongly Disagree ☐ Disagree ☐ Undecided ☐ Agree ☐ Strongly Agree

A-5 Using agile software development methods has improved project delivery commitments (on-time delivery of project functionality).

☐ Strongly Disagree ☐ Disagree ☐ Undecided ☐ Agree ☐ Strongly Agree

A-6 Using agile software development methods enables me to accomplish tasks more quickly.

☐ Strongly Disagree ☐ Disagree ☐ Undecided ☐ Agree ☐ Strongly Agree

A-7 It was easy for me to become skillful at using agile software development methods.

☐ Strongly Disagree ☐ Disagree ☐ Undecided ☐ Agree ☐ Strongly Agree

A-8 People who are important to me think that I should use agile software development methods.

☐ Strongly Disagree ☐ Disagree ☐ Undecided ☐ Agree ☐ Strongly Agree

A-9 I have the knowledge necessary to use agile software development methods.

☐ Strongly Disagree ☐ Disagree ☐ Undecided ☐ Agree ☐ Strongly Agree

A-10 Using agile software development methods increases my productivity.

☐ Strongly Disagree ☐ Disagree ☐ Undecided ☐ Agree ☐ Strongly Agree

A-11 I find agile software development methods easy to use.

☐ Strongly Disagree ☐ Disagree ☐ Undecided ☐ Agree ☐ Strongly Agree

A-12 Management has been helpful in using agile software development methods.

☐ Strongly Disagree ☐ Disagree ☐ Undecided ☐ Agree ☐ Strongly Agree

A-13 Agile software development methods are compatible with other types of software development methods that I use.

☐ Strongly Disagree ☐ Disagree ☐ Undecided ☐ Agree ☐ Strongly Agree

A-14 Using agile software development methods has reduced post-delivery defects (in production environments).

☐ Strongly Disagree ☐ Disagree ☐ Undecided ☐ Agree ☐ Strongly Agree

A-15 Using agile software development methods increases my chances of getting a pay raise.

☐ Strongly Disagree ☐ Disagree ☐ Undecided ☐ Agree ☐ Strongly Agree

A-16 In general, the organization has supported my use of agile software development methods.

☐ Strongly Disagree ☐ Disagree ☐ Undecided ☐ Agree ☐ Strongly Agree

A-17 A specific person (or group) is available for assistance with difficulties encountered when using agile software development methods.

☐ Strongly Disagree ☐ Disagree ☐ Undecided ☐ Agree ☐ Strongly Agree

A-18 Using agile software development methods has increased project team satisfaction.

☐ Strongly Disagree ☐ Disagree ☐ Undecided ☐ Agree ☐ Strongly Agree

A-19 Using agile software development methods has increased customer (end-user) satisfaction.

☐ Strongly Disagree ☐ Disagree ☐ Undecided ☐ Agree ☐ Strongly Agree

A-20 Using agile software development methods has increased project success rates (delivered versus cancelled projects).

☐ Strongly Disagree ☐ Disagree ☐ Undecided ☐ Agree ☐ Strongly Agree

Please answer the following 3 questions using these definitions of management: **Corporate Officer**- examples include the Chief Information Office or president of Information Technology. **Director** – level of management that is responsible for one or more departments. **Program Manager** – level of management that directly supervises Project Managers. **Project Manager** – level of management that directly supervises workers.

A-21 To the best of your knowledge, what is the highest level of management that sponsored/championed the adoption of agile software development methods?

☐ Corporate Officer

☐ Director

☐ Program Manager

☐ Project Manager

☐ Unknown or not

☐ Other, please specify: ____

A-22 To the best of your knowledge, what is the highest level of management that received training in agile software development methods?

☐ Corporate Officer

☐ Director

☐ Program Manager

☐ Project Manager

☐ Unknown or not

- ☐ Other, please specify: ____

A-23 To the best of your knowledge, what is the highest level of management to actually use agile software development methods?

- ☐ Corporate Officer
- ☐ Director
- ☐ Program Manager
- ☐ Project Manager
- ☐ Unknown or not
- ☐ Other, please specify: ____

A-24 To the best of your knowledge, which of the following best describes incentives or recognition used by your organization to promote adoption of agile software development?

- ☐ Individual Incentive or Recognition
- ☐ Team Incentive and Recognition
- ☐ Both Individual and Team Incentive or
- ☐ No Incentive or Recognition Provided
- ☐ Unknown or not
- ☐ Other, please specify: ____

A-25 Which of the following agile software development methods are currently used in your organization? (check all that apply)

- ☐ Adaptive Software Development (ASD)
- ☐ Crystal Methods
- ☐ Dynamic Systems Development Methodology (DSDM)
- ☐ Extreme Programming (XP)
- ☐ Feature Driven Development (FDD)
- ☐ Homegrown / Hybrid
- ☐ SCRUM
- ☐ WISDOM
- ☐ Other - please specify:____

A-26 Which of the following agile software development practices has your organization adopted? (check all that apply)

- ☐ 2-4 week iterations
- ☐ Active stakeholder participation
- ☐ Adaptive Project Management
- ☐ Agile coach
- ☐ Agile estimation & planning
- ☐ Automated testing
- ☐ Burn Down/Burn Up chart
- ☐ Collaborative acceptance testing
- ☐ Collective code ownership
- ☐ Co-located client
- ☐ Co-located project team
- ☐ Continuous code integration
- ☐ Customer focus group review
- ☐ Daily standup meeting
- ☐ Define requirements in small pieces
- ☐ Exploratory spikes
- ☐ Feature based planning
- ☐ Feature list
- ☐ Frequent releases
- ☐ Group design
- ☐ Incremental infrastructure
- ☐ Information radiators
- ☐ Iteration planning
- ☐ Lean project management training
- ☐ Mid-iteration acceptance test planning
- ☐ Pair programming

- ☐ Production quality acceptance test for every iteration
- ☐ Refactoring
- ☐ Regression testing
- ☐ Retrospective
- ☐ Self-Tasking teams
- ☐ Simple design
- ☐ Test driven design/development (TDD)
- ☐ Other - please specify: ____

A-27 To the best of your knowledge, how long has your organization been using agile software development methods?

- ☐ < 6 months
- ☐ 6 – 12 months
- ☐ 1 – 2 years
- ☐ 2 – 4 years
- ☐ > 4 years

A-28 Which of the following best describes the level of change to requirements or features for your current or most recently completed agile-based project?

- ☐ No changes on project
- ☐ Low (1-2 changes per week)
- ☐ Medium (3 – 5 changes per week)
- ☐ High (> than 5 changes per week)

A-29 Which of the following best describes the staffing environment for your current or most recently completed agile-based project?

- ☐ Stable (low staff turnover)
- ☐ Volatile (high staff turnover)

A-30 Which of the following best describes the number of team members on your current or most recently completed agile-based project?

- ☐ 1-2 people
- ☐ 3-5 people
- ☐ 6-10 people
- ☐ 11-15 people
- ☐ 16-20 people
- ☐ >20 people

A-31 Which of the following best describes the physical location of the team members on your current or most recently completed agile-based project?

- ☐ Co-located on building floor
- ☐ Co-located in same building or adjacent buildings
- ☐ Dispersed across same city or town
- ☐ Dispersed across geographic timeframes (different region or country)

A-32 Which of the following best describes the percent of your organization that uses agile software development methods? (estimate as close as possible)

- ☐ 5 % or less
- ☐ 10 %
- ☐ 25 %
- ☐ 50 %
- ☐ 75 %
- ☐ 100%
- ☐ Unknown

You are 75% complete . . . please select '**Section C: Organizational Characteristics.**

<u>Section B: Future Adoption</u>

To the best of your ability, please select one answer to each of the following statement that describe factors ***which may contribute to future adoption*** of agile software development methods in your organization:

B-1 I expect agile software development methods to be useful in my job.

☐ Strongly Disagree ☐ Disagree ☐ Undecided ☐ Agree ☐ Strongly Agree

B-2 I expect my interaction with agile software development methods would be clear and understandable.

☐ Strongly Disagree ☐ Disagree ☐ Undecided ☐ Agree ☐ Strongly Agree

B-3 I will use agile software development methods because people who influence my behavior think that I should use them.

☐ Strongly Disagree ☐ Disagree ☐ Undecided ☐ Agree ☐ Strongly Agree

B-4 I will use agile software development methods because resources (support team, training, infrastructure) necessary to use they will be available.

☐ Strongly Disagree ☐ Disagree ☐ Undecided ☐ Agree ☐ Strongly Agree

B-5 I expect agile software development methods will improve project delivery commitments (on-time delivery of project functionality).

☐ Strongly Disagree ☐ Disagree ☐ Undecided ☐ Agree ☐ Strongly Agree

B-6 I expect agile software development methods to enable me to accomplish tasks more quickly.

☐ Strongly Disagree ☐ Disagree ☐ Undecided ☐ Agree ☐ Strongly Agree

B-7 It should be easy for me to become skillful at using agile software development methods.

☐ Strongly Disagree ☐ Disagree ☐ Undecided ☐ Agree ☐ Strongly Agree

B-8 I will use agile software development methods because people who are important to me think that I should use them.

☐ Strongly Disagree ☐ Disagree ☐ Undecided ☐ Agree ☐ Strongly Agree

B-9 I will use agile software development methods because I will have the knowledge necessary to use them.

☐ Strongly Disagree ☐ Disagree ☐ Undecided ☐ Agree ☐ Strongly Agree

B-10 I expect agile software development methods to be easy to use.

☐ Strongly Disagree ☐ Disagree ☐ Undecided ☐ Agree ☐ Strongly Agree

B-11 I will use agile software development methods because management may be helpful to me in using them.

☐ Strongly Disagree ☐ Disagree ☐ Undecided ☐ Agree ☐ Strongly Agree

B-12 I expect agile software development methods to be compatible with other types of software development methods that I use.

☐ Strongly Disagree ☐ Disagree ☐ Undecided ☐ Agree ☐ Strongly Agree

B-13 I expect agile software development methods to reduce post-delivery defects (in production environments).

☐ Strongly Disagree ☐ Disagree ☐ Undecided ☐ Agree ☐ Strongly Agree

B-14 I expect agile software development methods to increase my chances of getting a raise.

☐ Strongly Disagree ☐ Disagree ☐ Undecided ☐ Agree ☐ Strongly Agree

B-15 I will use agile software development methods because the organization in general will support their use.

☐ Strongly Disagree ☐ Disagree ☐ Undecided ☐ Agree ☐ Strongly Agree

B-16 I will use agile software development methods because a specific person (or group) will be available for assistance with difficulties encountered when using them.

☐ Strongly Disagree ☐ Disagree ☐ Undecided ☐ Agree ☐ Strongly Agree

B-17 I expect agile software development methods to increase project team satisfaction.

☐ Strongly Disagree ☐ Disagree ☐ Undecided ☐ Agree ☐ Strongly Agree

B-18 I expect agile software development methods to increase customer (end-user) satisfaction.

☐ Strongly Disagree ☐ Disagree ☐ Undecided ☐ Agree ☐ Strongly Agree

B-19 I expect agile software development methods to increase project success rates (delivered versus cancelled projects).

☐ Strongly Disagree ☐ Disagree ☐ Undecided ☐ Agree ☐ Strongly Agree

You are 75% complete . . . please select '**Section C: Organizational Characteristics.**

Section C: Organizational Characteristics

C-1 Which of the following statements best describes the culture of your organization?

- ☐ My organization concentrates on having a high degree of flexibility and individuality.
- ☐ My organization concentrates on flexibility, concern for people, and sensitivity for customers.
- ☐ My organization focuses on internal stability and control which flows from a strict chain of command characterized by formalized procedures.
- ☐ My organization focuses on the external environment, is driven by results, and is very competitive.

C-2 Which of the following statement best describes the type of organizational learning orientation of your organization?

- ☐ Organizational learning involves doing things better without necessarily examining or challenging our underlying beliefs and assumptions. The goal is improvement and fixes that often take the form of procedures or rules.
- ☐ Organizational learning involves considering our actions in the framework of our operating assumptions. We change the way we make decisions and deepen the understanding of our assumptions. The goal is to make major fixes or changes, like redesigning an organizational function or structure.
- ☐ Organizational learning involves learning principles and extends beyond insight and patterns to context. We produce new commitments and ways of learning. The goal is to fundamentally change how the organization learns how to learn. As a result, we develop a better understanding of how to respond to our environment and deepen our comprehension of why we chose to do things the way we do.

C-3 In the context of improving software development processes, to the best of your knowledge which of the following areas does your organization plan to focus on in the next six months? (check all that apply)

- ☐ Improve ability to manage uncertainty and risk
- ☐ Improve client satisfaction overall
- ☐ Improve development team satisfaction overall
- ☐ Improve software development productivity
- ☐ Improve software quality
- ☐ Improve responsiveness to clients changing requirements
- ☐ Improve time to market
- ☐ Reducing software development cost
- ☐ We have no current plans to improve software development processes

You are 90% complete . . . click 'Section D: Demographic Questions'.

<u>Section D: Demographics</u>

D-1 In which of the following geographic region do you primarily work in?

- ☐ Africa
- ☐ Asia
- ☐ Central America
- ☐ Eastern Europe
- ☐ European Union
- ☐ Middle East
- ☐ North America
- ☐ Oceania
- ☐ South America
- ☐ The Caribbean

D-2 Which of the following best describes your organizations industry classification?

- ☐ Banking
- ☐ Consulting
- ☐ Education
- ☐ Financial Services (other than banking)
- ☐ Government
- ☐ Health Care
- ☐ Information Technology
- ☐ Insurance
- ☐ Manufacturing
- ☐ Retail
- ☐ Telecommunications/Internet Service Provider
- ☐ Utility
- ☐ Other - please specify:

____

D-3 How large is your Information Technology organization including all personnel involved with software development and support (estimate as close as possible)?

- ☐ < =5
- ☐ 6 - 25
- ☐ 26 - 50
- ☐ 51 - 100
- ☐ 101 - 250
- ☐ 251 - 500
- ☐ 501 - 1000
- ☐ Greater than 1000

D-4 Which of the following best describes your role in your organization?

- ☐ Agile Coach
- ☐ Architect
- ☐ Business Partner/IT User
- ☐ Consultant
- ☐ Chief Information Officer (CIO)
- ☐ Chief Technology Officer (CTO)
- ☐ Client Manager
- ☐ Developer
- ☐ Development Manager
- ☐ Director
- ☐ IT Staff
- ☐ Product Manager
- ☐ Project Manager
- ☐ Program Manager
- ☐ Quality Assurance
- ☐ Team Lead
- ☐ Tester
- ☐ Trainer
- ☐ Other - please specify:

D-5 How many years of software development experience do you have (estimate as close as possible)?

- ☐ 1-2 years
- ☐ 3-5 years
- ☐ 6-10 years
- ☐ 11-15 years
- ☐ > 15 years
- ☐ Not applicable

D-6 What is your age?

- ☐ Less than 30
- ☐ 30-39
- ☐ 40-49
- ☐ 50-59
- ☐ Greater than 59

D-7 What is your highest level of education?

- ☐ Less than high school
- ☐ High school or General Equivalency Diploma (GED)
- ☐ 2-year college degree
- ☐ 4-year college degree
- ☐ Masters degree
- ☐ Doctoral degree

D-8 What is your gender?

- ☐ Female
- ☐ Male

D9 Please provide any addition comments regarding this research if desired:

You are 100% complete . . . click ‘Submit’ to exit!

Thank you for participating in this survey!

APPENDIX B

SURVEY COVER LETTER

Dear Chairperson,

As an IT professional, you know the importance of relevant information when making informed decisions about software development methods best used to deliver projects.

I am a fellow colleague and university researcher investigating the topic of agile versus traditional software development method adoption to further increase the body of knowledge that organizations can use to make informed adoption decisions.

As such, during the month of September I am soliciting the global network of SPIN chapters to participate in an anonymous survey of IT practitioners to aid this research. Summary results will be provided to Chairpersons when compiled in November.

The survey should take no more than 10 minutes to complete and does not collect information attributable to a person or organization.

Please forward the survey link listed below to any of your members that wish to participate.

Thanks for your consideration!

Sincerely,

Tracy Lambert
Nova Southeastern University
tlambert@nova.edu

Dr. Richard McCarthy
Nova Southeastern University
rmccarth@nova.edu

Survey of Software Development Method Adoption

REFERENCES

Adams, D. A., Nelson, R. R., & Todd, P. A. (1992). Perceived usefulness, ease of use, and usage of information technology: A replication. *MIS Quarterly, 16*(2), 227-247.

Agerfalk, P.J. (2004). Investigating actability dimensions: A language/action perspective on criteria for information systems evaluation. *Interacting with Computer, 16*(5), 957-998.

Ajzen, I., & Fishbein, M. (1980). *Understanding attitudes and predicting social behavior*. Englewood Cliffs, NJ: Prentice-Hall.

Ajzen, I. (1985). From intentions to actions: A theory of planned behavior. *Springer Series in Social Psychology* (pp. 11-39). Berlin: Springer.

Ajzen, I. (1991). The theory of planned behavior. *Organizational Behavior and Human Decision Processes, 50*(2), 179-211.

Ambler, S. (2002). *Agile Modeling: Effective Practices for Extreme Programming and The Unified Process*. New York, NY: John Wiley and Sons.

Ambler, S. (2006). *Agile Adoption Rate Survey: March 2006.* Retrieved August 20, 2008, from http://www.ambysoft.com/scottAmbler.html.

Ambler, S. (2008). *Agile Adoption Rate Survey: February 2008.* Retrieved August 20, 2008, from http://www.ambysoft.com/scottAmbler.html.

Babbie, E. (2004). *The practice of social research.* Thomson Wadsworth.

Bagozzi, R. P. (2007). The legacy of the technology acceptance model and a proposal for a paradigm shift. *Journal of the Association for Information Systems, 8*(4), 243-254.

Barney, J. B. (1986). Types of competition and the theory of strategy: Towards an integrative framework. *Academy of Management Review, 11*, 791-800.

Barney, J. B. (1991). Firm resources and sustained competitive advantage, *Journal of Management, 17*(1), 99-120.

Barney, J.B., McWillianson, A. & Turk, T. (1989). *On the relevance of the concept of entry barriers in the theory of competitive strategy.* Paper presented at the annual meeting of the Strategic management Society, San Francisco, CA.

Beck, K. (1999). Embracing change with extreme programming. *IEEE Computer*, *32*, 70-77.

Becker, G.S. (1964). *Human Capital.* New York: Columbia.

Berinato, S. (2001). The secret to software success. *CIO*, *14*(18), 76-82.

Boehm, B. (2002). Get ready for agile methods, with care. *IEEE Computer*, *35*, 64-69.

Boehm, B., & Turner, R. (2003). *Balancing Agility and Discipline: A Guide for the Perplexed*. Reading, MA: Addison-Wesley.

Brightman, H. & Schneider, H. (1994). *Statistics for business problem solving.* Cincinnati, OH: South-Western.

Broadbent, M., Weill, P., & St. Clair, D. (1997). The implications of information technology infrastructure for business process redesign. *MIS Quarterly, 23*, 159-182.

Campbell, D.J. (1988). Task complexity: A review and analysis. *Academy of Management Review, 13*(1), *40-52.*

Caves, R.E. & Porter, M. (1977). From entry barriers to mobility barriers: Conjectural decisions and contrived deterrence to new competition. *Quarter Journal of Economics*, *91*, 241-262.

Cockburn, A. (2001). *Agile software development*. Boston: Addison-Wesley.

Cockburn, A., & Highsmith, J. (2001). Agile software development: The business of innovation. *Computerworld*, *34*, 120-122.

Crawford, S.D., Couper, M.P., & Lamias, M.J. (2001). Web surveys. *Social Science Computer Review*, *19*(2), 146-162.

Creswell, J. (1994). *Research design: qualitative and quantitative approaches*. Sage Publications.

Cronbach, L. J. (1951). Coefficient alpha and the internal structure of tests. *Psychometrika, 16*(3), 297-334.

Davis, F. D. (1989). Perceived usefulness, perceived ease-of-use, and user acceptance of information technology. *MIS Quarterly*, *13*(3), 319-339.

Davis, F. D., Bagozzi, R. P. & Warshaw, P. R. (1989). User acceptance of computer technology: A comparison of two theoretical models. *Management Science, 35*, 982-1003.

Davis, F. D., Bagozzi, R. P. & Warshaw, P. R. (1992). Extrinsic and intrinsic motivation to use computers in the workplace. *Journal of Applied Social Psychology*, *22*, 1111-1132.

Dishaw, M.T., & Strong, D.M. (1998). Extending the technology acceptance model with task-technology fit constructs. *Information and Management*, *36*, 9-21.

Dishaw, M.T., & Strong, D.M. (2003). The effect of task and tool experience on maintenance case tool usage. *Information Resources Management Journal*, *16*(3), 1-16.

Dishaw, M.T., Strong, D.M., & Brandy, D.B. (2006). Extending task technology fit with computer self-efficacy. *Database for Advances in Information Systems, 37*, 96-107.

Dyba, T. (2000). An instrument for measuring key factors of success in software process improvement. *Empirical Software Engineering*, *5*, 357-390.

Erickson, J., & Siau, K. (2003). Theoretical and practical complexity of unified modeling language: A Delphi study and metrical analyses. In proceedings of the international conference on information Systems, 183-194.

Erickson, J., Lyytinen, K. & Siau, K. (2005). Agile modeling, agile software development, and extreme programming: The state of research. *Journal of Database Management*, *16*, 88-100.

Fink, A. (2000). *How to Design Survey Studies*. Thousand Oaks, CA: Sage.

Fink, A. (2003). *How to Manage, Analyze, and Interpret Survey Data*. Thousand Oaks, CA: Sage.

Fishbein, M., & Ajzen, I. (1975). *Belief, attitude, intention, and behavior: An introduction to theory and research.* Reading, Mass., Don Mills, Ontario: Addison-Wesley.

Fitzgerald, B., Harnett, G., & Conboy, K. (2006). Customizing agile methods to software practices at Intel Shannon. *European Journal of Information Systems*, *15*, 200-213.

Fowler, M. (2002). The new methodology. Available at www.martinfowler.com/articles/newMethodology.com

Glass, R. (2003). Questioning the software engineering unquestionables. *IEEE Software*, *20*, 1-2.

Gong, M., Xu, Y., & Yu, Y. (2004). An enhanced technology acceptance model for web-based learning. *Journal of Information Systems Education*, *15*(4), 365-374.

Goodhue, D.L., & Thompson, R. L. (1995), Task-technology fit and individual performance, *MIS Quarterly, 19*(2), 213-236.

Grenning, J. (2001). Launching extreme programming at a process intensive company. *IEEE Software, 18(6),* 27-33.

Grover, V., Jeong, S.R., Kettinger, W. J., & Wang, S. (1995). The implementation of business process reengineering. *Journal of Management Information Systems*, *12*(1), 109-144.

He, Z., & Wong, P. (2004). Exploration vs. exploitation: An empirical test of the ambidexterity hypotheses. *Organizational Science, 15*, 481-494.

Herbsleb, J., & Moitra, D. (2001). Global software development. *IEEE Software*, *18*(2), 16-20.

Highsmith, J. (1999). *Adaptive software development: A collaborative approach to managing complex systems*. New York: Dorset House.

Hirsch, M. (2002). Making RUP agile. In Proceedings of SIG Programming Languages: OOPSLA, 1-8.

Hirshliefer, J. (1980). Price theory and applications. Englewood Cliffs, NJ: Prentice-Hall.

Holmstrom, H., Fitzgerald, B., & Agerfalk, P.J. (2006). Agile practices reduce distance in global software development. *Information Systems Management*, *23*, 7-18.

Hoogeweegen, M. R., Teunissen, W. J. M., Vervest, P. H. M., & Wagenaar, R. W. (1999). Modular network design: Using information and communication technology to allocate production tasks in a virtual organization. *Decision Sciences, 30*, 1073-1103.

Hovorka, D.S., & Larsen, K.R. (2006). Enabling agile adoption practices through network organizations. *European Journal of information Systems, 15*, 159-168.

Katila, R., & Ahuja, G. (2002). Something old, something new: A longitudinal study of search behavior and new product introduction. *Academy of Management Journal*, *45*(6), 1183-1194.

Kaye, B.K., & Johnson, T.J. (1999), Research methodology: Taming the cyber frontier. *Social Science Computer Review, 17*(3), 3230337.

Klopping, I. M., & McKinney, E. (2004). Extending the technology acceptance model and the task-technology fit model to consumer e-commerce. *Information Technology, Learning, and Performance Journal, 22,* 1, 35-48.

Kotlarsky, J., & Oshri, I. (2005). Social ties, knowledge sharing and successful collaboration in globally distributed system development projects. *European Journal of Information Systems, 14*, 37-48.

Larman, C. (2004). *Agile and iterative development: a manager's guide.* Boston: Addison-Wesley.

Larman, C., & Basili, V. (2003). Iterative and incremental development: A brief history. *IEEE Computing, 36*(6), 47-56.

Lippman, S. & Rumelt, R. (1982). Uncertain imitability: an analysis of interfirm differences in efficiency under competition. *Bell Journal of Economics, 13*, 418-438.

March, J. (1991). Exploration and exploitation in organizational learning. *Organizational Science, 2*, 71-87.

Martin, R. (2003). Extreme programming development through dialog. *IEEE Software, 17*, 12-13.

McCabe, R. (2006), *Disciplined Agility Standard*, Retrieved August 20, 2008 from http://www.software.org/AgileStandard.html.

McConnell, S. (1998). *Software project survival guide.* Redmond: Microsoft Press.

Morrel-Samuels, P. (2002). Getting the truth into workplace surveys. *Harvard business Review*, 80*(2),* 111-119.

Nerur, S., Mahapatra, R., & Mangalaraj, G. (2005). Challenges of migrating to agile methodologies. *Communications of the ACM, 48*, 73-78.

Norusis, M.J., (2009). *PASW Statistics 18*. Prentice hall.

O'Reilly, C.A., III & Tushman, M.L. (2004). The ambidextrous organization. *Harvard Business Review, 82*, 74-81.

Palmer, R.S., & Felsing, J.M. (2002). *A practical guide to feature-driven development*. Upper Saddle, NJ: Prentice-Hall.

Paulk, M. (2001). Extreme programming from a CMM perspective. *IEEE Software, 18*, 19-26.

Porter, M.E. (1980). *Competitive Strategy.* New York, NY: New York Free Press.

Porter, M.E. (1981). The contribution of industrial organization to strategic management. *Academy of Management Review, 6*, 609-620.

Porter, M.E. (1985). *Competitive Advantage*. New York, NY: New York Free Press.

Porter, M. E., (1990). *Why are firms successful*. Paper presented at the Fundamental Issues in Strategy Conference. Napa, CA.

Porter, M. E., (1996). What is strategy? *Harvard Business Review*. November-December, 61-78.

Porter, M. E., (1997). How competitive forces shape strategy, *Harvard Business Review*, July-August, 1-10.

Porter, M.E. (2001). Strategy and the Internet. *Harvard Business Review*, Vol. 79, 63-78.

Punter, T., Ciolkowski, M., Freimut, B., & John, I. (2003). Conducting on-line surveys in software engineering. In *Proceedings of the 2003 International Symposium on Empirical Software Engineering*. Washington DC: IEEE Computer Society.

Rihania, S. & Goyer, R. (2001). Complexity: An appropriate framework for development? *Progress in Development Studies, 1*(3), 237-245.

Rogers, E.M. (1995) *Diffusion of Innovations*. Free Press.

Royce, W. (1970). Managing the development of large software systems. *Proceeding of IEEE*, January, 1970.

Schneberger, S., Amoroso, D.L., & Durfee, A. (2007). Factors that influence the performance of computer-based assessments: An extension of the technology acceptance model. *The Journal of Computer Information Systems, 48*(2), *74-90.*

Schonlau, M., Fricker, R.D., & Elliot, M.N. (2000). *Conduction Research Surveys via E-Mail and the Web.* Santa Monica, CA: Rand.

Schaefer, D.R., & Dillman, D.A. (1998). Development of a standard e-mail methodology. *Public Opinion Quarterly, 62*(3), 378-397.

Schleyer, T. K. L., & Forrest, J. L. (2000). Methods for the design and administration of web-based surveys. *Journal of the American Medical Informatics Association*, 7(4), 416-425.

Schwaber, K., & Beedle, M. (2002). *Agile software development and Scrum*. Upper Saddle River, NJ: Prentice-Hall.

Schwaber, C., & Fichera, R. (2005). Corporate IT leads the second wave of agile adoption. Forrester, November 30, 2005. Retrieved August 20th, 2008 from http://www.forrester.com/Research/Document/Excerpt/072113833400.html.

Schwarz, A., & Chin, W. (2007). Looking forward: Toward an understanding of the nature of IT acceptance. *Journal of the Association for information Systems*, *8*(4), 230-232.

Sekaran, U. (2000). *Research methods for business*. New York: John Wiley & Sons.

Sheehan, K. B., & Hoy, M.G. (1999). Using e-mail to survey internet users in the United States: Methodology and assessment. *Journal of Computer-Mediated Communication, 4*(3).

Sheppard, B. H., Hartwick, J., & Warshaw, P.R (1988). The theory of reasoned action: A meta-analysis of past research with recommendations for modifications and future research. *Journal of Consumer Research*, *15*, 325-343.

Siau, K. (1999). Information modeling and method engineering: A psychological perspective. *Journal of Database Management*, *10*, 44-50.

Staples, S. D., & Seddon, P. (2004). Testing the technology-to-performance chain model. *Journal of Organizational and End User Computing, 16*(4), 17-36.

Stapleton, J. (1997). *DSDM, dynamic systems development method: The method in practice*. Harlow: England: Addison-Wesley.

Szajna, B. (1994). Software evaluation and choice: Predictive validation of the technology acceptance instrument. *MIS Quarterly*, *18*(3), 319-325.

Taylor, S., & Todd, P. A. (1995). Assessing IT usage: The role of prior experience. *MIS Quarterly*, *19*(4), 561-570.

Thomer, J.F. (1987). *Organizational capital: the path to higher productivity and well-being*. New York: Praeger

Van Alstyne, M. (1997). The state of network organizations, a survey of three frameworks. *Journal of Organizational Computing*, *7*, 83-152.

VersionOne (2008), 3rd Annual Survey: 2008 "State of Agile Development", Retrieved August 20, 2008, from http://www.versionone.com/pdf/3rdAnnualStateOfAgile_FullDataReport.pdf.

VersionOne (2009), 2009 Agile Practitioner Salary Survey, Retrieved January 4th, 2011 from http://www.aspe-sdlc.com/offers/pickups_0386572/2009_agile_salary_survey.pdf

Venkatesh, V. (2006). Where to go from here? Thoughts on future directions for research on individual-level technology adoption with a focus on decision making. *Decision Science, 37*(4), 497-518.

Venkatesh, V., Davis, F. D., & Morris, M., G. (2007). Dead or alive? The development, trajectory and future of technology adoption research, *Journal of The Association For Information Systems*, *8*(4), 267-286.

Venkatesh, V, Morris, M. G., Davis, G.B., Davis, F.D. (2003). User acceptance of information technology: Toward a unified view. *MIS Quarterly*, *27*(3), 425-478.

Venkatesh, V., & Speier, C. (1999). Computer technology training in the workplace: A longitudinal investigation of the effect of mood. *Organizational Behavior and Human Decision Processes, 79*, 1-28.

Venkatesh, V., Speier, C., & Morris, M. G. (2002). User acceptance enablers in individual decision making about technology: Toward an integrated model. *Decision Sciences, 33*(2), 297-316.

Vinekar, V., Slinkman, C.W., & Nerur, S. (2006). Can agile and traditional Systems development approaches coexist? An ambidextrous view. *Information Systems Management*, *23*, 31-42.

Williamson, O, (1975). *Markets and hierarchies.* New York Free Press

Wixom, B. H., & Todd, P. A. (2005). A theoretical integration of user satisfaction and technology acceptance. *Information Systems Research*, *16*(1), 85-102.

Wood, R.E. (1986). Task complexity: Definition of construct. *Organizational Behavior and Human Decision Processes, 37*(1), *60-82.*

Zahra, S.A, & George, C. (2003). Absorptive capacity: A review, reconceptualization and extension. *Academy of Management Review*, *27*, 185-203.

Zigurs, I., & Buckland, B. K. (1998). A theory of task/technology fit and group support systems effectiveness. *MIS Quarterly*, *22*(3), 313-334.

Zigurs, I., Buckland, B. K., Connolly, J. R., & Wilson, E.V. (1998). A test of task-technology fit theory for group support systems. *Database for Advances in Information Systems, 30*(3), 34-50.

Zipf, G. (1949). *Human behavior and the principle of least effort.* New York: Hafner Publishing Company.

BIBLIOGRAPHY

Agerfalk, P.J., Fitzgerald, B., Holmstrom, H., Links, B., Lundell, B., & O'Conchuir, E. (2005). A framework for Considering Opportunities and Threats in Distributed Software Development. In Proceedings of the International Workshop on Distributed Software Development (DISD), August 2005, Australian Computer Society.

Almutairi, H. (2007). Is the 'technology acceptance model' universally applicable?: The case of the Kuwaiti Ministries. *Journal of Global Information Technology Management, 10*(2), 57-80.

Amin. H. (2007). Extending the technology acceptance model for SMS banking: analyzing the gender gap among students. *International Journal of Business and Society, 8*(1), 15-27.

Baker-eveleth, L., Eveleth, D. M., O'Neill, M., & Stone, R. W. (2006). Enabling laptop exams using secure software: Applying the technology acceptance model. *Journal of Information Systems Education, 17*(4), 413-420.

Baloh, P. (2007). The role of fit in knowledge management systems: tentative propositions of the KMS design. *Journal of Organizational and End User Computing, 19*(4), 22-41.

Bhattacherjee, A., & Hikmet, N. (2008). Reconceptualizing organizational support and its effect on information technology usage: Evidence from the health care sector. *The Journal of Computer Information Systems, 48*(4), 69-76.

Boehm, B. (1985). A spiral model for software development and enhancement. Proceeding of an international workshop on software process and software environments, March, 1985.

Borjesson, A., Martinsson, F., & Timmeras, M. (2006). Agile improvement practices in software organizations, *European Journal of Information Systems, 15*, 169-182.

Chakrabarty, S., Whitten, D., & Green, K. (2007). Understanding service quality and relationship quality in is outsourcing: Client orientation & promotion, project management effectiveness, and the task-technology-structure fit. *The Journal of Computer Information Systems, 48*(2), 1-15.

D'ambra, J., & Wilson, C. S. (2004). Explaining perceived performance of the world wide web: Uncertainty and the task-technology fit model. *Internet Research, 14*(4), 294-310.

Davis, F. D. (1986). A technology acceptance model for empirically testing new end-user information systems: Theory and results. (Doctoral dissertation, Sloan School of Management, Massachusetts Institute of Technology).

Davis, F. D., Bagozzi, R. P., & Warshaw, P. R. (1989). User acceptance of computer technology: A comparison of two theoretical models. *Management Science, 35*(8), 982-1003.

Derrick, V., Neufeld, J., Ong, L., & Higgins, C. (2007) Charismatic leadership and user acceptance of information technology. *European Journal of Information Systems, 16*(4), 494-510.

Dinev, T., & Hu, Q. (2007). The centrality of awareness in the formation of user behavioral intention toward protective information technologies. *Journal of The Association for Information Systems, 8*(7), 386-408.

Elwood, S., Changchit, C., & Cutshall, R. (2006). Investigating students' perceptions on laptop initiative in higher education: An extension of the technology acceptance model. *Campus - Wide Information Systems, 23*(5), 336-349.

Eriksson, C. I., Kalling, K., Åkesson, M., & Fredberg, T. (2008). Business models for m-services: Exploring the e-newspaper case from a consumer view. *Journal of Electronic Commerce in Organizations. 6*(2), 29-57.

Fetscherin, M., & Lattemann, C. (2008). User acceptance of virtual worlds. *Journal of Electronic Commerce Research, 9*(3), 231-242.

Fusilier, M., & Durlabhji, S. (2005). An exploration of student internet use in India: The technology acceptance model and the theory of planned behavior. *Campus - Wide Information Systems, 22*(4), 233-246.

Gane, C., & Sarson, T. (1980). Structured Methodology: What have we learned? *Computerworld*, *14*(38), 52-57.

Gilb, T. (1988). *Principles of software engineering management.* Addison-Wesley.

Goodhue, D.L. (1995), Understanding user evaluations of information systems. *Management Science*, *41*(12), 1827-1844.

Hwang, Y. (2005). Investigating enterprise systems adoption: Uncertainty avoidance, intrinsic motivation, and the technology acceptance model. *European Journal of Information Systems*, *14*(2), 150-161.

James, T., Pirim, T., Boswell, K., Reithel, B., & Barkhi, R. (2006). Determining the intention to use biometric devices: An application and extension of the technology acceptance model. *Journal of Organizational and End User Computing, 18*(3), 1-24.

Lallmahamood, M. (2007). An examination of individual's perceived security and privacy of the internet in Malaysia and the influence of this on their intention to use e-commerce: Using an extension of the technology acceptance model. *Journal of Internet Banking and Commerce, 12*(3), 1-26.

Lee, H., Fiore, A. M., & Kim, J. (2008). The role of the technology acceptance model in explaining effects of image interactivity technology on consumer responses. *International Journal of Retail & Distribution Management, 34*(8), 621-644.

Lu', J., Yu, C. S., Liu, C., & Yao, J. (2003). Technology acceptance model for wireless Internet. *Internet Research, 13*(3), 206-222.

Ma, Q., & Lu, L. (2004). The technology acceptance model: A meta-analysis of empirical findings. *Journal of Organizational and End User Computing, 16*(1), 59-72.

Martin, R. (2003). *Agile software development: principles, patters, and practices*. Upper Salle River, NJ: Prentice-Hall.

Masrek, M. N., Karim, S., A. & Hussein. R. (2007). Investigating corporate intranet effectiveness: A conceptual framework. *Information Management & Computer Security, 15*(3), 168.

Mccloskey, D. (2003). Evaluating electronic commerce acceptance with the technology acceptance model. *The Journal of Computer Information Systems, 44*(2), 49-57.

Mccloskey, D. W. (2006). The importance of ease-of-use, usefulness, and trust to online consumers: An examination of the technology acceptance model with older consumers. *Journal of Organizational and End User Computing, 18*(3), 47-65.

Mccoy, S., Everard, A., & Jones, B. M. (2005). An examination of the technology acceptance model in Uruguay: A focus on culture. *Journal of Global Information Technology Management, 8*(20), 27-45.

Mckechnie, S., Winklhofer, H., & Ennew, C. (2006). Applying the technology acceptance model to the online retailing of financial services. *International Journal of Retail & Distribution Management, 34*(5), 388-410.

Meso, P., & Jain, R. (2006). Agile software development: Adaptive systems principles and best practices. *Information Systems Journal, 23*, 19-30.

Miller, K. (2005). *Communications theories: perspectives, processes, and contexts*. New York: McGraw-Hill.

Moore, G.A. (1991). *Crossing the chasm: marketing and selling high-tech goods to mainstream customers*. Harper Business.

Ndubisi, N. O., Gupta, O. K., & Ndubisi, G. C. (2005). The moguls' model of computing: integrating the moderating impact of users' persona into the technology acceptance model. *Journal of Global Information Technology Management*, *8*(1), 27-47.

Ndubisi, N. O., & Jantan, M. (2003). Evaluating is usage in Malaysian small and medium-sized firms using the technology acceptance model. *Logistics Information Management*, *16*(6), 440-450.

Peffers, K. (2006). Models for technology acceptance, culture-based it products, software reuse libraries, and digital broadcasting. *Journal of Information Technology Theory and Application, 8*(2), 3-5.

Perez, M., Sanchez, A. M., Carnicer, P. D., & Jimenez, M. J. V. (2004). A technology acceptance model of innovation adoption: The case of teleworking. *European Journal of Innovation Management*, *7*(4), 280-291.

Pikkarainen, T., Pikkarainen, K., Karjaluoto, H., & Pahnila. S. (2004). Consumer acceptance of online banking: An extension of the technology acceptance model. *Internet Research, 14*(3), 224-235.

Rao, S., & Troshani, I. (2007). A conceptual framework and propositions for the acceptance of mobile services. *Journal of Theoretical and Applied Electronic Commerce Research*, *2*(2), 61-73.

Sahu, G. P., & Gupta, M. P. (2007). Users' acceptance of e-government: A study of in Indian central excise. *International Journal of Electronic Government Research, 3*(3), 1-21.

Savitskie, K., Royne, M. B., Persinger, E. F., Grunhagen, M., & Witte, C., L. (2007). Norwegian internet shopping sites: an application & extension of the technology acceptance model. *Journal of Global Information Technology Management*, *10*(4), 54-73.

Sheppard, B. H., Hartwick, J., & Warshaw, P.R (1988). The theory of reasoned action: A meta-analysis of past research with recommendations for modifications and future research. *Journal of Consumer Research*, *15*, 325-343.

Silva, L. (2007). Post-positivist review of technology acceptance model, *Journal of The Association for Information System*, *8*(4), 255-266.

Simon, S. J., & Paper, D. (2007). User acceptance of voice recognition technology: An empirical extension of the technology acceptance model. *Journal of Organizational and End User Computing*, *19*(1), 24-50.

Slyke, C. V., Ilie, V., Lou, H., & Stafford, T. (2007). Perceived critical mass and the adoption of a communication technology. *European Journal of Information Systems, 16*(3), 270-283.

Standish Group (1994). The CHAOS report. Retrieved February 17, 2008 from sample research from the Standish Group web site: http://www.standishgroup.com/sample_research/PFDpages/chaos1994.pdr

Standish Group (1999). The CHAOS report. Retrieved February 17, 2008 from sample research from the Standish Group web site: http://www.standishgroup.com/sample_research/PFDpages/chaos1999.pdr

Standish Group (2001). The CHAOS report. Retrieved February 17, 2008 from sample research from the Standish Group web site: http://www.standishgroup.com/sample_research/PFDpages/extreme_chaos.pdf

Standish Group (2003). *CHAOS Chronicles Version 3.0.* Yarmouth, MA: The Standish Group international.

Standish Group (2004). CHAOS demographics and project resolution. Retrieved February 17, 2008 from sample research from the Standish Group web site: http://www.standishgroup.com/sample_research/PFDpages/q3-spotlight.pdf.

Straub, D. W., & Burton-jones, A. (2007). Veni, vidi, vici: breaking the TAM logjam. *Journal of the Association for Information Systems, 8*(4), 223-229.

Uzoka, F. E., Abiola, R. O., & Nyangeresi, R. (2008). Influence of product and organizational constructs on ERP acquisition using an extended technology acceptance model. *International Journal of Enterprise Information System 4*(2), 67-84.

Venkatesh, V., Davis, F.D. (2002). A theoretical extension of the technology acceptance model: Four longitudinal field studies. *Management Science*, *46*(2), 186-204.

Yao, Y., & Murphy, L. (2007). Remote electronic voting systems: An exploration of voters' perceptions and intention to use. *European Journal of Information Systems, 16*(2), 106-120.

Yi, Y., Wu, Z., & Tung, L. L. (2005). How individual differences influence technology usage behavior? Toward an integrated framework. *The Journal of Computer Information Systems, 46*(2), 52-63.

Yourdon, E. (1976). The emergence of structured analysis. *Computer Decisions*, *5,* 58-59.

Yourdon, E. (1977). Trends in software science. *Journal of Systems Management*, *5*, 39-41.