An Agile Enterprise Architecture Driven Approach to Enhance Communication in Geographically Distributed Agile Development

Author:

Yehia Ibrahim Alzoubi

Supervisors:

Dr Asif Gill

Dr Bruce Moulton

Dr Ahmed Al-Ani

A thesis submitted in fulfilment of the requirements for the degree of

Doctor of Philosophy

Faculty of Engineering and Information Technology

University of Technology Sydney

Date of Submission:

December 2016

Certificate of Original Authorship

I, Yehia Alzoubi, certify that the work in this thesis has not previously been submitted for a degree nor has it been submitted as part of requirements for a degree except as fully acknowledged within the text. I also certify that the thesis has been written by me. Any help that I have received in my research work and the preparation of the thesis itself has been acknowledged. In addition, I certify that all information sources and literature used are indicated in the thesis.

Yehia Alzoubi

December 2016

Acknowledgements

First, and foremost, praise is due to Allah who has bestowed upon me many blessings and patience, ability, and skills to complete this work.

I would like to express my sincere thankfulness to all those (named or not) who helped me to complete this long trip. I would like to thank my supervisors Drs. Asif Gill, Ahmed Al-Ani, and Bruce Moulton for their supportive guidance.

Special thanks go to my supervisor Dr. Asif Gill for his continuous support and encouragement, without which this study would have never been completed. I am also appreciative of his countless hours of advice and unlimited support in improving my research skills.

I would like to thank each respondent and interviewee who has participated in this study. I also would like to thank each expert who has helped me in evaluating the research outcomes.

I am grateful to my brothers, sisters (especially Om-Bishr), relatives, father and mother in law, and friends who have always encouraged me during this time.

I would like to thank my wife for her patience and continuous encouragement to complete this thesis. My last words are for my two little angels: Yousef and Judi, whose childhood play and mess have been pleasant melodies to my ears and eyes during my trip.

Dedication

I dedicate this work to my late father (1933-1992), may Allah bless his soul, who first taught me the alphabets of this life. Even after all these years, you are in my thoughts every day dad. I also dedicate this work to my unlimited love and support-my mum (A'ishah), may Allah give you healthiness and long life.

List of Publications

Book Chapters

Alzoubi, Y.I. & Gill, A.Q. 2016, 'An agile enterprise architecture-driven model for geographically distributed agile development', in D. Vogel, X. Guo, H. Linger, C. Barry, M. Lang & C. Schneider (eds), *Transforming Healthcare Through Information Systems*, Springer International Publishing, pp. 63-77.

Journal Articles

Alzoubi, Y.I., Gill, A.Q. & Al-Ani, A. 2015, 'Distributed agile development communication: an agile architecture driven framework', *Journal of Software*, vol. 10, no. 6, pp. 681-94. {Double-blind review; ERA ranked B journal}

Alzoubi, Y.I., Gill, A.Q. & Al-Ani, A. 2016, 'Empirical studies of geographically distributed agile development communication challenges: a systematic review', *Information and Management*, vol. 53, no. 1, pp. 22-37. {Double-blind review; ERA ranked A* journal}

Conference Proceedings

Alzoubi, Y.I. & Gill, A.Q. 2014, 'Agile global software development communication challenges: a systematic review', *In Proceedings of the 18th Pacific Asia Conference on Information Systems (PACIS 2014)*, p. 20. {Double-blind review; ERA ranked A Conference}

Alzoubi, Y.I. & Gill, A. 2015, 'An agile enterprise architecture driven model for geographically distributed agile development', *In Proceedings of the 24th International Conference on Information System Development (ISD 2015)*, Harbin, China. {Double-blind review; ERA ranked A Conference}

Table of Contents

List of Figures

List of Tables

List of Abbreviations

AMOS	Analysis of Moment Structures
ASD	Agile Software Development
CB-SEM	Covariance-Based Structural Equation Model
CFA	Confirmatory Factor Analysis
DSD	Distributed Software Development
DSDM	Dynamic Software Development
EA	Enterprise Architecture
EFA	Exploratory Factor Analysis
EM	Expectation Maximisation
FDD	Feature Driven Development
GDAD	Geographically Distributed Agile Development
HTMT	Hetero-Trait Mono-Trait
IS	Information System
IT	Information Technology
KSD	Kanban Software Development
LISREL	Linear Structural Relations
LSD	Lean Software Development
MAR	Missing Data at Random
MCAR	Missing Completely at Random
MIS	Management of Information System
MTMM	Multi-Trait Multi-Method
PLS	Partial Least Squares
RQ	Research Question
SD	Software Development
SEM	Structural Equation Modelling
SLDC	Systems Development Life Cycle
SLR	Systematic Literature Review
SPSS	Statistical Package for the Social Science
TDD	Test Driven Development
VAF	Variance Accounted for
VIF	Variance Inflation Factor
XP	Extreme Programming

Glossary

Item	Explanation	Reference
Active Communication	Refers to the process of exchanging information between agile team members (i.e. senders and receivers) formally and informally, which should be efficient and effective.	Alzoubi & Gill 2015
Agile Software Development (ASD)	A software development method is said to be an agile software development method when a method is mainly people focused and communication-oriented, flexible (ready to adapt to expected or unexpected change at any time), speedy (encourages rapid and iterative development of the product in small releases), lean (focuses on shortening timeframe and cost and on improved quality), responsive (reacts appropriately to expected and unexpected changes), and learning (focuses on improvement during and after product development).	Qumer & Henderson-Sellers 2006b, p. 125
Agility	A persistent behaviour or ability of a sensitive entity that exhibits flexibility to accommodate expected or unexpected changes rapidly, follows the shortest time span, uses economical, simple and quality instruments in a dynamic environment and applies updated prior knowledge and experience to learn from the internal and external environment.	Qumer & Handerson-Sellers 2006b, p. 281
Agile Enterprise Architecture (EA)	A blueprint that describes the overall structural, behavioural, social, technological, and facility elements of an enterprise's operating environment that share common goals and principles with the ability of responsiveness (scans, senses and reacts appropriately to expected and unexpected changes), flexibility (adapts to expected or unexpected change at any time), speediness (accommodates expected or unexpected changes rapidly), leanness (focuses on	Gill 2013a,

	reducing waste and cost without compromising on quality), and learning (focuses on enterprise fitness, improvement and innovation).	
Communication	A process in which participants create and share information with one another in order to reach a mutual understanding.	Rogers 1986, p. 199
Communication Challenges	Refer to the characteristics of each medium that decrease communication efficiency and effectiveness.	Clark & Brennan 1991
Communication Effectiveness	Delivering a message to the receiver who understands it as it was intended with minimal disruption and misunderstanding, even if it takes a long time.	Alzoubi & Gill 2016
Communication Efficiency	Delivering a message to a receiver with high quality and with minimal time, cost, effort, and resources required to establish communication.	Alzoubi & Gill 2016
Enterprise Architecture	The organising logic for business processes and IT infrastructure, reflecting the integration and standardisation requirements of the company's operating model.	Ross et al. 2006, P.9
Formal Communication	Refers to explicit and clear communication such as the agile requirements backlog, plans, and card walls.	Herbsleb & Mockus 2003
Informal Communication	Refers to the communication that happens outside the official reporting hierarchy of the team.	Herbsleb & Mockus 2003
Software Development (SD)	The tasks undertaken to construct a [software-based product], and the management of this effort, by a group of stakeholders with agendas, who engage in transactions over time within an institutional context by applying structure to their work with a set of tools and methodologies, and who judge outcomes of their efforts and act accordingly.	Sambamurthy & Kirsch 2000, p. 400
Software Functionality	The extent to which the delivered software system meets its functional goals, user needs, and technical requirements.	Lee & Xia 2010

Software Quality	Represents the assessment of the quality of the task performed by the individual, or the pair, on the programming task. It is reflective of how well the code satisfied the requirements stipulated in the problem statement and produced correct results.	Balijepally et al. 2009, p. 102
Software Performance	Refers to four dimensions; on-time completion (meets its baseline goals for duration), on-budget completion (meets its baseline goals for cost), functionality (meets its functional scope goals), and quality (a good working product).	Alzoubi & Gill 2016

Abstract

Agile development is a highly collaborative environment, which requires active communication among stakeholders. This active communication helps in producing high quality working software systems in short releases and iterations. Due to the ever-increasing competition, there is an increasing interest among practitioners and researchers in contemporary geographically distributed agile development (GDAD). GDAD claims to offer several benefits over co-located agile development such as lower production cost, around the clock development, faster time to market, and the liberty of involving the most talented developers across the globe. However, in the GDAD environment, active communication is difficult to achieve due to many challenges such as differences in geographical locations and time.

Literature has reported that agile enterprise architecture (EA) could help enhancing GDAD communication and performance. However, little empirical evidence is known to support this claim. Furthermore, it is not clear how to effectively achieve and study active communication construct in GDAD in terms of its dimensions, determinants and effects on performance? As a result, there is a lack of understanding about how GDAD organisations can establish and maintain active communication among distributed teams. This dissertation contributes to this research gap, first, by developing a research model based on an extensive systematic literature review on the GDAD communication challenges, techniques and strategies to mitigate these challenges, and the impact of communication on GDAD software performance. This study provides important insights about GDAD communication by identifying and empirically examining the relationships among the two dimensions of active communication (communication efficiency and communication effectiveness), one antecedent that can be controlled (agile enterprise architecture (EA)), and four aspects of GDAD performance (on-time completion, on-budget completion, software functionality, and software quality).

The study then validates the research model using an integrated research approach that combines quantitative and qualitative data analyses. The quantitative data are collected using a survey technique from 160 responses and analysed using Partial Least Squares

(PLS) analyses. The qualitative data are collected using interview techniques through 10 post hoc case studies and analysed using content analysis technique.

This study reports that agile EA has positive impacts on communication efficiency and communication effectiveness, and on GDAD performance. It has also been found that communication efficiency and communication effectiveness have significant differential impacts on GDAD performance aspects. While communication efficiency is, generally, related to on-time and on-budget completions, communication effectiveness is, generally, related to functionality and quality aspects.

While the prior GDAD literature offers little guidance for GDAD communication issue, this research contributes to both theory and practice, and offers a number of useful insights and agile EA driven GDAD model. From theory perspective, insights and model are theoretically based on and empirically tested about the value and positive impact of agile EA on active communication dimensions and GDAD performance, and the impact of communication efficiency and communication effectiveness on GDAD performance in the GDAD environment. Moreover, from practice perspective, this study indicates that agile EA, communication efficiency, and communication effectiveness together increase the GDAD performance and thus, facilitate a better GDAD performance than in GDAD that does not employ agile EA.

Despite the above-mentioned contributions, like any other studies, this study has also some limitations such as sample size, time and potential analysis bias of applied qualitative and quantitative research methods. A number of steps were taken to mitigate or minimise the effects of these limitations. Thus, findings of this work should be considered with its limitaiotns when interpreting it in the relevant theoretical and practical context.

Chapter 1 Introduction

This dissertation is about investigating the impact of agile enterprise architecture (EA) on communication and software project performance, and the impact of communication on software project performance in geographically distributed agile development (GDAD) environment. This chapter begins by discussing the research background in Section 1.1. The research problem is discussed in Section 1.2. The research questions and objectives are discussed in Section 1.3. Section 1.4 outlines the research design and methodology. Section 1.5 outlines the research scope. A summary of the findings of this study and implications are presented in Section 1.6 and Section 1.7, respectively. Section 1.8 provides an overview of the remaining chapters of this thesis. Finally, Section 1.9 summarises this chapter.

1.1 Research Background

Agile software development (ASD) is a group of software development methods (approaches) or frameworks that promote adaptive planning, time-boxed iterative and incremental development, encourages rapid and flexible response to change, and requirements and solutions evolve through collaboration between cross-functional self-organised teams (Qumer & Henderson-Sellers 2006b). Agile approaches depend heavily on face-to-face communication and coordination among co-located team members and customers (Qumer & Henderson-Sellers 2006b).

Communication is defined as the process of exchanging information between senders and receivers (McQuail 1987). Rogers (1986) defines communication as the process in which participants create and share information with one another in order to reach a mutual understanding. These definitions draw our attention to the importance and effectiveness of communication between the parties included in agile development. However, agile methods promise faster development thus improving the communication

efficiency too (Fitzgerald et al. 2006; Pikkarainen et al. 2008). Agile Manifesto (the cornerstone of the agile development) states that the "most efficient and effective method of conveying information to and within a development team is face-to-face conversation" (Agile Manifesto 2001). Hence, this study refers to communication efficiency and communication effectiveness as the two dimensions of active communication (Alzoubi & Gill 2016; Gill 2015b). Accordingly, this study defines communication efficiency as delivering a message to a receiver with high quality and with minimal time, cost, effort, and resources required to establish communication. It also defines communication effectiveness as delivering a message to the receiver who understands it as it was intended with minimal disruption and misunderstanding, even if it takes a long time. These definitions are discussed in greater details in Ch.4.

Agile development practices focus on using informal active communication among the team members such as the agile requirements backlog, plans and card walls etc. (Qumer & Henderson-Sellers 2006b). Informal communication can be defined as personal, interactive and peer-oriented communication (Korkala & Maurer 2014; Smith et al. 1994). Also, it can be defined as the communication that takes place outside the official structure and without the knowledge of management (Herbsleb & Mockus 2003), which seems helpful in quickly identifying and auctioning issues and risks. Informal communication helps in filling and correcting mistakes quickly, which supports and ensures agile practices and principles (Agile Manifesto 2001).

Since agile approaches depend heavily on face-to-face active communication and coordination among co-located team members and customers, physical proximity is essential for participants to engage in informal communication (Ramesh et al. 2006). This kind of communication, in the co-located and local context, results in saving time and effort, and reducing documentation, which increase the benefits for business and enhances customer satisfaction (Qumer & Henderson-Sellers 2008). The success of agile development in co-located small and medium size teams motivates large software organisations to scale and adopt agile approaches in a distributed or global environment. The combination of geographically distributed development and agile practices, known as ''geographically distributed agile development'' (GDAD), seems to offer many benefits, such as low production cost, the opportunity to involve the most talented

developers around the world and faster time to market (Abrahamsson et al. 2009). Forrester estimated that approximately 3.3 million jobs and $136 billion in salaries were expected to shift overseas by 2015. Around one million of those jobs are in IT-related fields (Engardio et al. 2003).

GDAD refers to the agile development that includes teams or/and team members distributed over different locations and time zones (Ramesh et al. 2006). Accordingly, GDAD teams or team members can be locally distributed in different physical locations within the same country, or can be globally distributed around the globe in different time-zones or different countries (Mishra et al. 2012; Ramesh et al. 2006).

However, applying agile approaches in GDAD is not straightforward and possesses many challenges, especially communication related challenges that can impact project performance (Agerfalk et al. 2009; Sarker & Sarker 2009). It has been argued that delivering a GDAD project takes 2.5 times more than the same project in the co-located local agile environment (Herbsleb & Mockus 2003). Korkala and Maurer (2014) argued that communication related issues are the root for all other GDAD challenges. A recent survey by Scott Ambler on scaling agile shows that, more than 50% of GDAD projects have failed (Ambler 2012a). More recently, the Standish Group has released CHAOS report that covers the last five years (2011-2015). The Standish Group identified project failure through meeting three critical elements: on-time, on-budget, and satisfaction (The Standish Group 2015). The results show that 23% of the software projects that used GDAD had failed comparing to 4% that used co-located agile development and 59% of GDAD comparing to 38% of co-located had been challenged to deliver the final outcomes (The Standish Group 2015). Another survey shows that the greater is the level of geographic distribution, the greater is the risk due to communication and coordination challenges and the lower success rate (Ambler 2012b). Despite these challenges, it is not practical to assume that every project can or should be delivered in a co-located local agile environment. Thus, the need for active communication in GDAD environment among team members and customers is very important and even more critical than in co-located agile development in order to overcome the uncertainty and changeable customer's requirements, and to achieve the highest software performance (Agile Manifesto 2001; Alzoubi & Gill 2016).

1.2 Research Problem

It has been reported in literature that active communication may enhance GDAD design and quality by reducing the project development time and cost (Paasivaara & Lassenius 2014), but the mechanism of how active communication can do that has not been reported. Hence, to a large extent, research on how active communication can be achieved in GDAD environment is limited. In spite of a number of tool oriented solutions and strategies (Alzoubi et al. 2016) that have been introduced to mitigate GDAD active communication challenges (e.g., using available tools and technologies such as Wiki, teleconference, videoconference, and exchanging visits), there are still unresolved concerns or challenges such as differences in personal attitudes, languages and cultures. These challenges decrease the efficiency and effectiveness of GDAD communication and can not be solved by using the available tools. Therefore, there is still a pressing need for discovering alternative solution options (Korkala & Maurer 2014).

Most recently, agile EA has been proposed as an alternative solution to help enhancing GDAD communication and performance (Ambler 2014b; Gill 2015b). EA refers to organising of business processes and IT infrastructure that reflects the integration and standardisation requirements of the organisation's operating model (Ross et al. 2006). Agile EA applies agile principles and focuses on collaboratively and incrementally developing, adapting and sharing information about business and IT elements, their relationships to each other and the overall enterprise environment to effectively guiding the implementation of a number of projects (Ambler 2014b; Gill 2015b; Korhonen et al. 2016). It has been suggested that the holistic and integrated views of agile EA offer the "possibility to see and discuss how different parts (the ICT systems, the processes, etc.) are interconnected and interplay. Understanding means not only knowing what elements the enterprise consists of and how they are related from different aspects, but also how the elements work together in the enterprise as a whole" (Karlsen 2008, p. 219). Agile EA seems appropriate to support GDAD, however, little empirical evidence is known about the impact of agile EA on GDAD communication and performance. Furthermore, in the context of GDAD communication and performance, it is not clear how to effectively achieve and study active communication construct in GDAD in terms of its

dimensions, determinants and effects on performance (Korkala et al. 2009; Pikkarainen et al. 2008). The above-mentioned gaps drive this research project, which is particularly concerned with the study of agile EA and its impact on GDAD active communication and performance. This research problem is further discussed in the next section.

1.3 Research Questions and Objectives

The aforementioned gaps provide the justification for conducting this research study which **aims to** enhance GDAD communication by answering the following main research question:

How to enhance communication in geographically distributed agile development?

To address this main question, there is a need to build a clear picture of the GDAD active communication issue. Therefore, this study first systematically reviewed the challenges of GDAD communication and identifies a number of challenges and the solutions/ techniques or strategies to mitigate the impact of these challenges. Hence, two subordinate questions were investigated, which are:

RQ1: What are the challenges or factors that limit GDAD communication?

RQ2: Which strategies, techniques or practices have been used to overcome these challenges and enhance GDAD communication?

Among the identified strategies, using EA in agile development was seen to have potential for enhancing GDAD active communication. To explain the impact of agile EA on GDAD active communication, this study puts forward the third question:

RQ3: How does agile EA affect GDAD active communication?

Moreover, since EA is related to deliver high performance in traditional software development, it was important to study the extent to which agile EA may impact the performance in GDAD environment. Therefore, the fourth question was formulated:

RQ4: How does agile EA affect GDAD software performance?

Furthermore, to understand the extent to which active communication is behind delivering high performance software in GDAD we formulated the fifth question:

RQ5: How does GDAD active communication affect GDAD software performance?

Finally, in order to extract the implications for practitioners in GDAD industry we formulated the sixth question:

RQ6: How can the findings of this research assist GDAD practitioners in overcoming GDAD communication issues?

Hence, the specific objectives of this research are to:

- Review the GDAD communication literature and identify the challenges and limitations of GDAD active communication.
- Identify the strategies, techniques or practices that have been used to enhance GDAD active communication as well as the impact of active communication on GDAD performance.
- Review the agile EA literature and explain the strategic role of agile EA in enhancing GDAD active communication and GDAD performance.
- Develop and examine a unique theoretical model depicting the influence of agile EA on GDAD active communication and GDAD performance, and the influence of active communication on GDAD performance using relevant theory.
- Define the constructs that constitute the conceptual model and develop measures to operationalise them.
- Collect data from teams' members whose organisations have employed agile EA driven GDAD.
- Empirically test the conceptual model using the collected data.

- Generate insights into the GDAD communication issue by providing some explanations for the findings, and suggesting some recommendations and directions for future research.

1.4 Research Design and Methodology

The approach taken to this research problem, as foreshadowed earlier is to start with what has already been published in the academic literature on the GDAD communication challenges and techniques to mitigate these challenges. This includes empirical or industrial experience-based literature. This was done by conducting a systematic literature review (SLR) and developing a candidate set of propositions on GDAD communication challenges and the current considerations to mitigate these challenges in real world projects. The main research iterations (including research activity, research technique, and research question) are summarised in Table 1.1. A detailed description of the research design and methodology is presented in Ch.5.

Table 1.1: Main research iterations in the study

Iteration	Activity	Method/Technique	Question
1	Identify challenges and limitations	Systematic Literature Review (SLR)	RQ1
2	Identify strategies, techniques or practices	SLR	RQ2
3	Create literature-based research model	SLR and literature review	RQ3-RQ5
4	Preliminary model evaluation, and measurement validation	Qualitative approach: preliminary field semi-structured interviews Quantitative approach: pilot tests of the survey instrument	RQ3-RQ5
5	Survey data collection	Quantitative approach: web-based and email-based survey	RQ3-RQ5
6	Measurement validation	Quantitative analysis: SPSS and PLS software packages	RQ3-RQ5
7	Hypotheses testing	Quantitative analysis: PLS software package	RQ3-RQ5
8	Post hoc case studies	Qualitative approach: field semi-structured interviews	RQ3-RQ5
9	Interpreting the findings	Writing the thesis: recommendations, contributions, implications	RQ6

The SLR results and literature review are used to create an initial literature-based research model that can then be empirically validated. The research model overview is shown in Figure 1.1. The central construct of the research model is GDAD active communication. Hypothesis 1 posits that agile EA has a positive effect on GDAD active communication (RQ3). Hypothesis 2 posits that agile EA has a positive effect on GDAD software performance (RQ4). Finally, hypothesis 3 posits that GDAD communication has a positive effect on GDAD software performance (RQ5). A detailed description of the research model and hypotheses is presented in Ch.4.

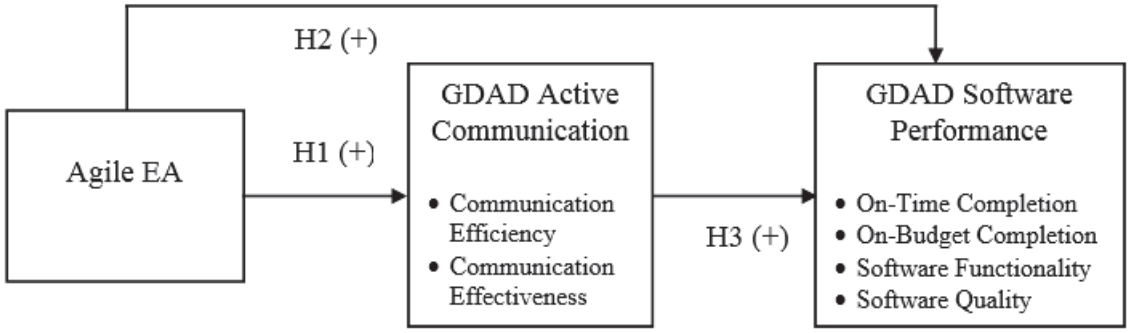


Figure 1.1. Research model (an overview diagram)

It is essential to choose the appropriate research method to identify the steps followed in answering the research questions (Leedy & Ormrod 2010). The combination of quantitative and qualitative approaches has received high attention in IS/IT field (Myers 1997). This combination helps in gaining better understanding of the research phenomenon by addressing limitations for both quantitative and qualitative approaches by providing the objectivity of the statistics and deeper understanding of the study context (Kaplan & Duchon 1988). Therefore, this study uses a combination of quantitative and qualitative approaches to validate the theoretical-based research model.

1.4.1 Research Strategy

This study is carried out by employing a sequential embedded research strategy (Creswell 2009). The sequential strategy refers to using different methods in sequence such that each method feeds the next method (Creswell 2009). This study uses a mixed-method approach. Primarily, the research follows a positivist paradigm and adopts a quantitative approach using survey as the research method. Quantitative research has been the dominant approach adopted in IS/IT research (Straub et al. 2005). Quantitative research allows studying a phenomenon without influencing or being influenced by it (Straub et al. 2005). Quantitative research enables the researcher to test relationships between the research model constructs and provide evidence to support the research hypotheses by employing objective measurements to collect research evidence (Walsham 1995). Secondly, to validate the findings of the survey, the research follows an interpretivist paradigm and adopts a qualitative approach using case studies as the research method. Qualitative research has increased during the last two decades (Pare 2004). Qualitative research helps in discovering new relationships in a complex

phenomenon (Pare 2004). It provides deeper explanations and results and reveals the complex dynamics of tensions among the research model constructs (Gable 1994). It also allows for comparisons of the theoretical results with the actual practice (Venkatesh et al. 2013).

The initial research model is evaluated by a group of five experts (three agile EA driven GDAD practitioners and two academics). The research model is then evaluated by conducting a pilot study that helps in validating the instruments used to measure the research model constructs. The measurement analysis includes testing content validity, measurements reliability, and convergent validity (correlation between variables). Content validity refers to the extent to which the indicators capture the major facets of the construct (Hair et al. 2014) and it is validated by the group of experts, as above. Reliability determines the extent to which the items in each variable are related to each other (Straub et al. 2004) and it is tested by Cronbach's alpha coefficient. Convergent validity checks if the instrument is measuring what it is supposed to measure (Straub et al. 2004) and it is assessed using factor analysis.

1.4.2 Survey

The quantitative part of this study is conducted by using a survey (using web-based SurveyMonkey tool and email-based techniques) method. The unit of analysis is a team member in an agile EA driven GDAD organisation. Moreover, the survey employs a comment part to qualitatively solicit opinions from respondents about the relationships between agile EA, active communication, and software performance in GDAD environment.

The collected data from the survey is analysed in two stages. In the first stage, the assessment and validation of the measurement model involves assessment of the reliability, convergent validity, and discriminant validity, as pre-conditions for testing the overall research model. After the measurement model is validated, the structural model is analysed for evidence supporting the theoretical model in the second stage. The assessment of the structural model validity involves testing the model's predictive power or variance explained (R^2), the effect size (f^2), the predictive relevance validity

by testing the cross-validated redundancy (Q^2), and the path coefficient (β) significance between constructs.

1.4.3 Semi-Structured Interviews

The qualitative part of this study is conducted by using case study (i.e. using semi-structured interviews technique) method to collect the data. Purposeful sampling is used to select 10 post hoc case studies of team members of organisations that employ agile EA driven GDAD. The choice of cases in this study is also opportunistic based on industry relationships initiated by the researcher and supervisors.

The collected data from the semi-structured interviews and the comments from the survey are analysed using content analysis as a non-automated technique. However, to provide a level of validation to the non-automated analysis technique and to have more confidence in the reliability of the analysis results, the transcribed data are parsed together through an automated content analysis using Leximancer 4.5 (QSR International 2016).

1.4.4 Theoretical Foundation

Theory is defined as a body of knowledge (Gregor 2006) or the description that intends to explain or predict a phenomenon (Weber 2003). Theoretical foundation provides an explanation to the employed theories in the research (Gregor 2006). According to Gregor (2006) classification, this study can be identified as having a prediction (i.e. provides predictions to a testable propositions) and explanation (provides explanations to a phenomenon) nature. While the survey data try to predict the relationships in the research model (i.e. between agile EA, active communication, and software performance), the interviews data provide in-depth explanation for the research model constructs and relationships.

While there are no theories that can be firmly associated to this thesis focus, Common Ground theory (Clarke & Brennan 1991) is used to conceptualise the active communication process dimensions (i.e. communication efficiency and communication

effectiveness). It is also used to conceptualise the role of agile EA views that increase the common ground which enhance active communication among GDAD teams and team members. Common ground refers to the mutual, common or joint knowledge, beliefs, and suppositions between two communicating parties (Clarke & Brennan 1991). Moreover, as it was discussed in Section 1.3, the impact of agile EA and active communication is investigated on the software project performance. The relationships between agile EA and active communication with software performance are established according to the available literature.

1.5 Research Scope and Key Assumptions

The scope and limitations of this study are manifested in the research problem and its units of analysis. The scope of this study is set with an explicit reference to agile EA-driven GDAD organisations. Given the specific aim and objectives of this study, the scope is focused on investigating the impact of using agile EA (as a holistic shared vision) on GDAD active communication and performance, and the impact of active communication on GDAD performance. The population considered in scope are the GDAD team members. The research data is limited to a sample population of 160 respondents from all around the world.

Using agile EA would not provide a specific communication tool. Indeed, it is used with the available communication tools (e.g., instant messaging, video-conference, email) to facilitate and enhance the communication in the GDAD environment. Thus, this study is not restricted to specific tools. In other words, agile EA artefacts can be communicated or shared via any social communication tools. Moreover, as discussed earlier, the focus of this research is to study agile EA as a whole including both integrated business and technology views. Partial business or technology view is not aligned with the research objectives of this study and therefore, no attempt has been made to study the individual views of agile EA and their impact on GDAD. This is because the individual GDAD project or solution architectures include a subset of different integrated business and technology views of the agile EA. Hence, the study of individual views of agile EA is not the aim of this research.

Several key assumptions are made for conducting this study. Firstly, the information distilled from the literature is accurate and represent valid and reliable research. Secondly, the questions asked are suitable measures for the focus of this study. Thirdly, the techniques used to collect and analyse data are appropriate for this type of research. Fourthly, the results achieved from the analysis are entirely accurate to the best of the researcher's knowledge. Finally, the conclusions drawn, with respect to the objectives and justifications of this study, are rigorous to be considered by GDAD organisations towards employing agile EA to enhance their communication process.

1.6 Research Findings

The research findings revealed that RQ3, RQ4, and RQ5 were supported from both quantitative and qualitative findings. Firstly, using agile EA was statistically found to have a very large impact on communication efficiency and a large impact on communication effectiveness. Agile EA explained 33.7% of the variance of communication efficiency and 31.7% of the variance of communication effectiveness. The qualitative findings also showed that agile EA can enhance both communication efficiency and communication effectiveness in GDAD environment.

Secondly, using agile EA was statistically found to have large impacts on both software functionality and software quality, medium impact on on-budget completion, and no impact on on-time completion. The qualitative findings also showed, in general, that agile EA enhance all performance aspects. However, some trade-off between on-time and on-budget with quality or functionality may occur sometimes. That is, while software functionality and software quality can be enhanced using agile EA, they may come at the price of on-time or on-budget completions in GDAD environment.

Finally, the relationship between active communication and delivering high software performance was statistically supported to a large extent. However, communication efficiency and communication effectiveness had different impacts on software performance. While communication efficiency had very large impacts on on-time completion and on-budget completion, communication effectiveness had very large impacts on delivering high software functionality and quality. The qualitative findings

showed large positive impacts of both communication efficiency and communication effectiveness on delivering high software performance in GDAD environment.

1.7 Research Implications

This research has some implications for academia, practitioners, teaching and learning, and the intellectual community. For academia, this research opens up the 'black box' of GDAD communication process and introduces two new agile development communication dimensions: communication efficiency and communication effectiveness. The impacts of these dimensions were empirically tested on GDAD software performance. Further research is needed to these dimensions in GDAD and related IS fields. Moreover, the research empirically showed evidence about the role of agile EA on enhancing both communication and software performance in GDAD environment, which opens the doors for further research to study the role of agile EA on other aspects of the GDAD such as legal compliance and customer complaints. Finally, the research has developed new measures for agile EA that can be used in future related research.

For practitioners, this research provides empirical evidence that GDAD communication can be enhanced by sharing agile EA views among GDAD teams without necessarily increasing the number of visits between distributed sites or introducing additional communication tools or technology. Moreover, this research provides empirical evidence that GDAD software performance will be enhanced by using the agile EA views. Practitioners need to pay more attention to agile EA and train their staff on using its artefacts. Furthermore, this research provides more insights to the communication process by introducing two dimensions of active communication (i.e. communication efficiency and communication effectiveness), and providing an empirical evidence of their different impacts on GDAD software performance. Therefore, practitioners need to be aware of the multifaceted nature of GDAD active communication and understand the tension between its two different dimensions in order to build appropriate types of GDAD communication.

For teaching and learning, this study increases the learning experience by investigating agile EA and Common Ground theory in the context of GDAD communication. This

also provides an opportunity to use the findings of this study and develop a new course to be taught to academics andto train practitioners on the value of employing agile EA on GDAD communication and performance or to improve the existing courses such as Enterprise Software Architecture and Middleware and Enterprise Architecture Practice courses which are taught in the University of Technology Sydney.

1.8 Organisation of this Thesis

This thesis, illustrated in Figure 1.2, is organised as follows:

- Chapter 2 provides the first part of the related literature review. It begins by reviewing ASD, in general. Then, it provides a comprehensive background about GDAD. Finally, it discusses the gaps in GDAD literature and the motivations for this research.

- Chapter 3 provides the second part of the review of related literature, focusing on GDAD communication challenges and the strategies, techniques and practices to mitigate these challenges. This chapter presents two reviews: the first SLR covers the GDAD challenges and the second SLR covers GDAD communication. Moreover, it provides the rationale of using agile EA as an enhancer to GDAD communication.

- Chapter 4 discusses the theoretical foundation of the research, and the development of the research literature-based model from literature review and the SLR in Ch.2 and Ch.3. The constructs of the research model are identified, and hypotheses are developed on the relationships between the constructs. The chapter discusses the three model constructs (i.e. agile EA, communication process, and software performance) in distributed environment, to position the study within the context of prior research. It identifies the common perspectives with respect to relationships between these three constructs and their sub-constructs.

- Chapter 5 discusses the research methodology that is used to gather the data for validating the proposed theoretical model developed in Ch.4. The chapter begins

with a discussion of the research design, and then discusses the philosophical foundation and the justification of methods used to collect data in this study. The chapter then discusses the first phase of collecting data by using a survey method, the evaluation of the model constructs by experts, the validation of measurements used by conducting a pilot study, and the quantitative data analysis approach used. Then, it discusses the second phase of collecting data by using the case study method; the semi-structures data collecting technique, the interview process and protocol, and the qualitative data analysis approached used. Finally, this chapter discusses the ethical considerations applied to collect both quantitative and qualitative data.

- Chapter 6 presents a discussion of the process of preparation and examination of the survey collected data. It starts by analysing the demographics of the research sample collected from the survey. Then, it discusses the data screening, missing data analysis, and outlier's tests. Testing for underlying assumptions (i.e. normality, linearity, multicollinearity, and homoscedasticity test) for reflective constructs are discussed also. Finally, it discusses the exploratory factor analyses (EFA) and generalisability test (i.e. common method bias and non-response bias) for reflective constructs.

- Chapter 7 discusses the survey findings. It discusses the measurement model validity (i.e. purifying the measurement scales) and the structural model validity (i.e. testing for construct validity). It discusses the different tests that were conducted to validate the measurement involving reliability, convergent validity, and discriminant validity. The chapter also discusses the research hypotheses testing through the validation of the full structural model using PLS estimation (using R^2, f^2, Q^2, and β values), and reports upon the research findings. The chapter then discusses the results of the second-order analysis of the structural model (i.e. software performance as a second-order latent construct).

- Chapter 8 discusses the case studies findings. It begins by discussing the case study description. Then, it discusses the process of validating and analysing the qualitative data collected. The chapter then discusses the findings of analysing the semi-

structured interviews of the post hoc case studies. The aim of these semi-structured interviews is to supplement the findings of the survey and provide deeper insights to the unsupported hypotheses.

- Chapter 9 provides an extensive discussion to the findings of the study, from both survey and case study methods, in the context of the available literature. The chapter then provides some recommendations for using agile EA and establishing communication in GDAD environment in order to deliver higher performance software projects. Finally, the chapter revalidate the research model according to the cross-sectional analysis of the findings from both survey and case studies.

- Chapter 10 concludes this thesis. The chapter begins by detailing the study's contributions. It then discusses the study's limitations and the opportunities of future research arising from this study. Finally, concluding remarks are given, which concludes both the chapter and the thesis.

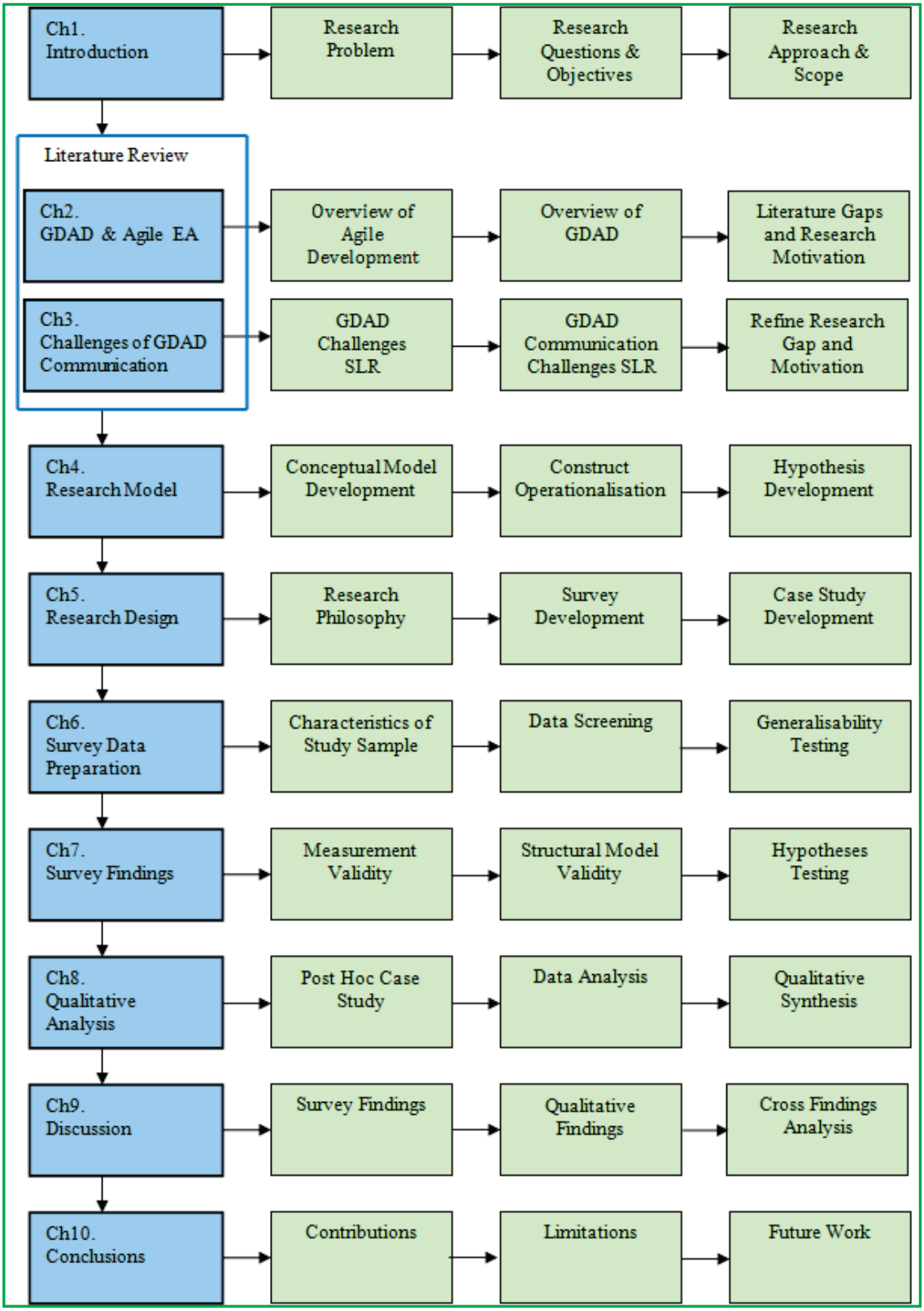


Figure 1.2. Structure of the thesis

1.9 Chapter Summary

This chapter outlined the foundation for the thesis investigation and the findings of this research. The research background and problem were firstly discussed. Then, research questions and objectives, research design, and research scope were outlined. Then, research findings and implications were summarised. Finally, the thesis structure was presented. The following chapter provides a context about GDAD. The background information discusses the traditional software development in co-located and in distributed contexts. It also discusses ASD and GDAD in greater detail. Moreover, the chapter critically identifies and discusses the gaps in previous literature regarding GDAD communication.

Chapter 2 Overview of Geographically Distributed Agile Development

2.1 Introduction

This chapter, drawn from academic and practitioner sources, reviews the first major streams of literature that is necessary for establishing the foundation of the study. The first stream of literature focuses on traditional software development (SD) and Agile SD (ASD) methods. The second stream focuses on distributed software development (DSD), in general, and GDAD, specifically. The purpose of the review is twofold:

- to provide an overview of the reference disciplines about agile approaches, communication in GDAD environment, and
- to identify the gaps in the existing GDAD communication base that this project seeks to address.

The chapter is structured as follows. It begins with an historical review of the software development evolving stages, which is presented in Section 2.2. A review of the literature on the traditional plan-driven and Spiral software developments are then presented in Section 2.3 and Section 2.4, respectively. Then ASD overview is discussed in Section 2.5. This section introduces and provides understandings on the historical development, prior definitions of ASD as well as the differences between ASD and plan-driven development. It also provides an overview of the most common ASD methods used. Section 2.6 outlines the concept and benefits of the traditional DSD. From this foundation, the review further focuses on GDAD which is presented in Section 2.7. Then, GDAD literature gaps are identified and summarised in Section 2.8 before the chapter is summarised in Section 2.9.

2.2 Historical Perspective on Software Development

Software development is defined as:

> "the tasks undertaken to construct a [software-based product], and the management of this effort, by a group of stakeholders with agendas, who engage in transactions over time within an institutional context by applying structure to their work with a set of tools and methodologies, and who judge outcomes of their efforts and act accordingly" (Sambamurthy & Kirsch 2000, p. 400).

Before investigating ASD methods, it is helpful to have a brief look into the history of SD. A brief historical overview of systems development methodologies has been provided by Avison and Fitzgerald (2003), who identified three eras to methodology: pre, early, and post-methodology. Avison and Fitzgerald (2003) have identified seven categories which emerged in the methodology era: structured, data-oriented, prototyping, object-oriented, participative, strategic, and systems.

1. Pre-methodology era: the focus in this era is on programming and technical problem-solving such that no formalisation is made to the development method.

2. Early methodology era: in this era, the systems development life cycle (SLDC) or plan-driven method are seen where the emphasis is on completing the pre-defined phases such as planning, analysis, design, implementation and maintenance. Although SDLC has become the base for many development projects, it contains numerous limitations such as failure to meet user needs and satisfaction, inflexibility, instability, and high maintenance costs.

3. Post-methodology era: in this era, the emphasis is on practitioners and researchers who adapt and use the previous methodologies' concepts, capabilities and strengths to provide a more effective methodology. According to the above categories, ASD fall into post-methodology era.

Figure 2.1 shows the evolvement of software development over the years. While plan-driven and DSD development have been used for many years, ASD and GDAD have recently gained more attention. For some organisations, "the problem is not the concept of a methodology, but the inadequacy of the current methodologies, prompting them to keep looking for different and better ones" (Avison & Fitzgerald 2003, p. 81). Therefore, to overcome the plan-driven SD and DSD shortfalls, as discussed in Section 2.3 and Section 2.4, parallel interest has been growing in the use of ASD methods and GDAD as potential solutions, as discussed in Section 2.5. Incremental method was suggested to replace the plan-driven methods since it allows addressing the customer's requirements and continuous enhancement to the system developed during the development life cycle. These two characteristics are also shared by ASD.

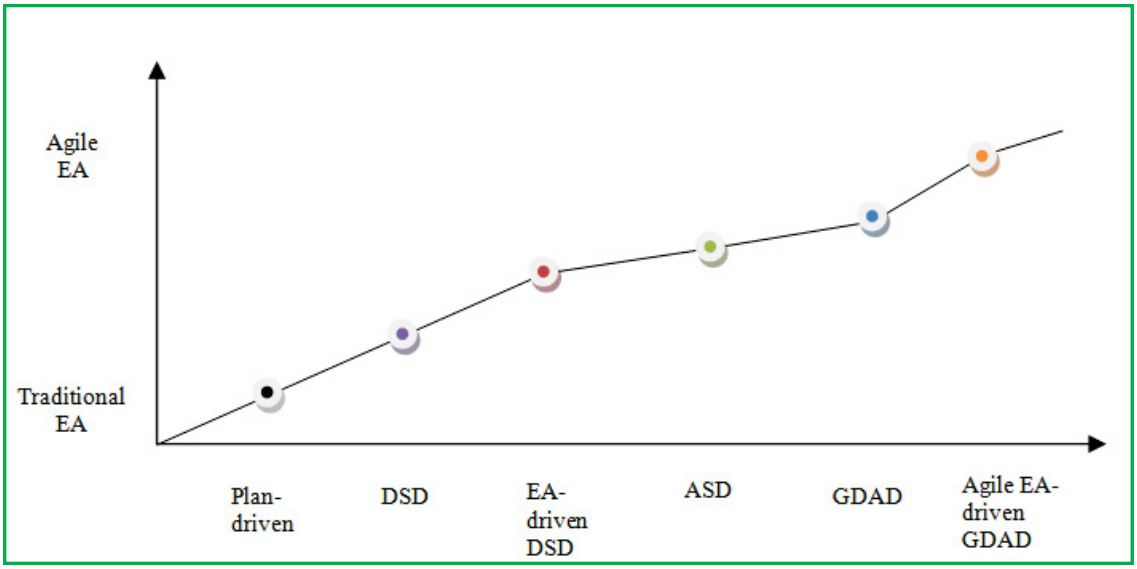


Figure 2.1. Software methods evolvement

In the past GDAD was considered inapplicable (Van Waardenburg & Van Vliet 2013).

Therefore, teams have to follow the standard traditional DSD process in GDAD, so it was suggested to tailor agile method in large different cultures enterprise (Van Waardenburg & Van Vliet 2013). However, as evidenced by the literature, GDAD has lucrative benefits for addressing issues such the over-budget, behind schedule, and inadequate to meet the needs of users. Moreover, the evolvement has shifted from using traditional EA to more agile EA, which is the focus of this study, which is expected to

be widely employed in the next few years (Gill 2015b). The different methods shown in Figure 2.1 will be discussed in the following sections.

Recently, the Standish Group has released CHAOS report 2015 (The Standish Group 2015). The CHAOS report studied 50,000 projects around the world, ranging from small to large systems implementations. The Standish Group identified project failure as the measurement of unfavourably meeting three critical elements: on-time, on-budget, and satisfaction (The Standish Group 2015). Table 2.1 summarises the outcomes of ASD and plan-driven (Waterfall) development projects over the last five years (2011-2015).

In comparison between ASD and the Waterfall methods, the report shows that across all project sizes agile approaches resulted in more successful projects and less outright failures such that 39% of the successful developed projects used ASD, while only 9% used Waterfall. Moreover, for large projects, the report shows that 18% of the successful projects used ASD, while only 3% used Waterfall. In addition, the results show that smaller projects have a much higher likelihood of success than larger ones, as shown in this table.

Table 2.1: Agile vs. Waterfall software development resolution (The Standish Group 2015)

Size	Method	Successful	Challenged	Failed
All size projects	Agile	39%	52%	9%
	Waterfall	11%	60%	29%
Large size projects	Agile	18%	59%	23%
	Waterfall	3%	55%	42%
Medium size projects	Agile	27%	62%	11%
	Waterfall	7%	68%	25%
Small size projects	Agile	58%	38%	4%
	Waterfall	44%	45%	11%

2.3 Plan-Driven (Waterfall/Traditional) Development

Plan-driven, Waterfall, or traditional SD development are names which refer to and describe the SD before developing the ASD. We will refer to all these names as plan-driven SD throughout the thesis.

Plan-driven development has been widely used since its creation due to the structured systems engineering approach used in it. However, all requirements must be clearly defined before development begins in effective plan-driven development (Avison & Fitzgerald 2003). This means that plan-driven development requires a well-planned, predictable, sequential SD process. Figure 2.2 represents the typical phases of plan-driven development.

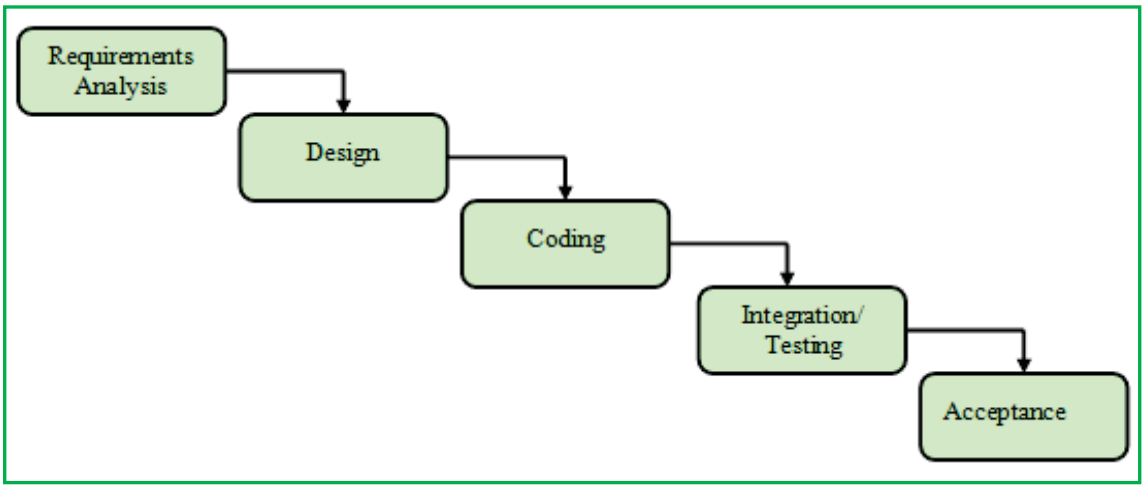


Figure 2.2. The plan-driven development model

Plan-driven method follows a systematic approach of converting requirements to user functionality through comprehensive functional analysis, design, coding, integration/ testing, and acceptance. This means that such actions are predicated on an all-inclusive collection of requirements. After getting the requirements, the development process and project cost structure are facilitated by an effectual work plan. Project cost can be normalised if the development process is stable (Boehm & Turner 2003). Generally, the plan-driven method provides greater insights into cost (Boehm & Turner 2003). Nevertheless, the plan-driven method has numerous disadvantages such as: (1)

misunderstanding/ mishandling customer requirements, (2) lack of customer feedback during development, (3) the limited ability to change the requirements during development, and (4) high maintenance cost (Avison & Fitzgerald 2003). Moreover, freezing the requirements for the entire duration of the project leads to a large accumulation of both the external and internal changes and results in project failures (Avison & Fitzgerald 2003).

Furthermore, plan-driven method emphasises providing very detailed documentation and well defined work packages. This results in one-way communication within the development team, which means a lack of consideration for the crucial two-way communications between team members (Boehm & Turner 2003). The impact of communication between team members is not critical to the production quality software during plan-driven development lifecycle; however, the stability, flexibility, and meeting customers' needs during the development lifecycle is essential and plan-driven lacks these virtues.

The shortfall of the plan-driven development in responding to stakeholders (i.e. customers and users) requirements has prompted practitioners and researchers to keep looking for more active development methodologies (Avison & Fitzgerald 2003). Hence, Spiral SD (engaged with prototyping) and ASD methods were introduced to provide a convenient method for stakeholders.

2.4 Spiral Software Development

Both plan-driven and agile projects begin with the initial understanding of the customer's requirements, which are essential to design and adequately build specifications. However, customer's requirements can be changed during development and the development method should be able to respond accordingly. Prototyping has been used as an approach to mitigate changeable requirements risks (Boehm 1988).

The engagement of prototypes with spiral software development, as shown in Figure 2.3, has been used as a well-accepted approach to mitigate the adverse impacts of unpredictable requirements on on-time completion, on-budget completion, and software

performance. Spiral development is described as the method that is explicitly considered as a risk-driven approach in software development (Boehm 1988). Spiral development was formulated to respond to evolving delivery and dynamically changing customer's requirements through delivering working prototypes and reducing risk. Spiral development provides software incremental, which is assumed as a defining principle for agile (Boehm & Turner 2003). Although Spiral development allows requirements change and decreases the risk, practitioners kept looking for more effective development methodologies that provide easy models or frameworks and enable customer's involvement in the development process. Therefore, they came up with what is known as ASD.

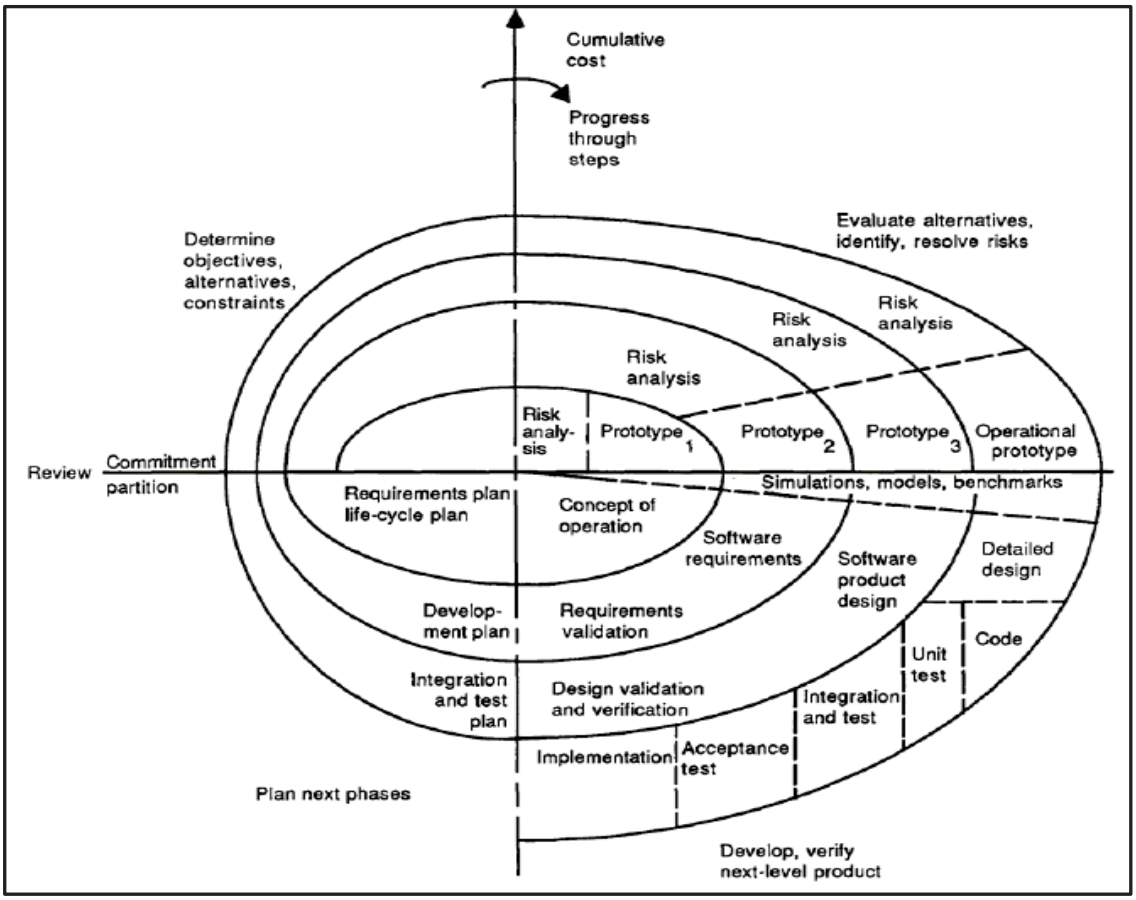


Figure 2.3. Spiral software development model (Boehm 1988)

2.5 Agile Software Development

The environment of the current business is described with changing requirements, emerging technologies, cost-effective and quick delivery (Cockburn 2007; Herbsleb & Moitra 2001). In addition, the high failure rate of the projects developed using plan-driven methods has promoted the SD industry to search for a substitute or new approach to satisfy the current business environment (Avison & Fitzgerald 2003). One of the solutions which were proposed to deal with the current business environment has been the creation of new SD methods called "Agile". Highsmith and Cockburn (2001) contend that agile methods "view change from a perspective that mirror today's turbulent business and technology environment" (p. 120).

The term "agile methods" has arisen when a group of scholars and practitioners met in 2001 to establish a common ground for SD methodologies. They come up with a statement entitled the "Manifesto for Agile Software Development". This statement summarises the major values of the ASD and established the twelve guiding principles shown in Table 2.2 (Agile Manifesto 2001).

Agile values:

Individuals and interactions; over processes and tools. This means that the most important success factor in software development is the people. They and their communication with each other should carry a much larger role than the traditional emphasis on coding knowledge and development tools.

Consumable solutions; over comprehensive documentation. Although software documentation is important, information transfer is more effective through the code itself and through people interaction.

Stakeholder collaboration; over contract negotiation. Successful SD requires frequent communication and collaboration between the customer and the developer, which is more important than a contract or a statement of work.

Responding to change; over following a plan. An adaptive short-term plan that evolves during SD is more flexible in responding to change and it is considered more effective than a long-term "up-front" project plan that is not adaptable.

Table 2.2: Agile software development principles (Agile Manifesto 2001)

Principle Description	
1.	Our highest priority is to satisfy the customer through early and continuous delivery of valuable software
2.	Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.
3.	Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale
4.	Business people and developers must work together daily throughout the project.
5.	Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done
6.	The most efficient and effective method of conveying information to and within a development team is face-to-face conversation
7.	Working software is the primary measure of progress.
8.	Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely
9.	Continuous attention to technical excellence and good design enhances agility
10.	Simplicity -the art of maximising the amount of work not done- is essential
11.	The best architectures, requirements, and designs emerge from self-organising teams
12.	At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behaviour accordingly

According to agile principles, ASD can be described as an incremental, simple, adaptive and cooperative development (Qumer & Handerson-Sellers 2006b). In a general sense, ASD concentrates on producing small iterative releases. These releases are described with high agility and cooperated with all stakeholders. This high cooperation needs a high level of communication, so ASD is a communication-oriented method. Hence, ASD is defined as:

> "a software development method is said to be an agile software development method when a method is mainly people focused and communication-oriented, flexible (ready to adapt to expected or unexpected change at any time), speedy (encourages rapid and iterative development of the product in small releases), lean (focuses on shortening timeframe and cost and on improved quality), responsive (reacts appropriately to expected and

> unexpected changes), and learning (focuses on improvement during and after product development)." (Qumer & Henderson-Sellers 2006b, p. 125)

Qumer & Handerson-Sellers (2006b) argue that any SD system can be assumed agile, partly agile or non-agile by applying the five agility attributes (i.e. flexibility, speed, leanness, learning and responsiveness) to the development method. In other words, agile development can be expressed to any SD that applies all or some of these attributes.

Agile is an iterative SD (Larman 2004). Iterative development refers to building a system or part of system efficiently within a short time period (Boehm & Turner 2003). The iterative progress allows customer to instantly incorporate feedback into the development process to improve system functionality (Larman 2004). Moreover, ASD is an integrated development and has co-located team members that hasten to understand, analyse, and incorporate timely feedback into the ASD iterations (Boehm & Turner 2003). The ASD starts with the initial understanding of customer's requirements (i.e. wants and desires) [Larman 2004]. As customer's specifications evolve, the ASD development team is provided with continuous feedback from the customer that results in incorporation of new features to the system (Boehm & Turner 2003). This process usually involves face-to-face synchronous communication within ASD team and with customers (Larman 2004). This communication style adds more cost reduction related to requirements change due to greater precision in communications within development teams (Larman 2004). The feedback enables ASD development team members to flexibly incorporate and appropriately respond to volatile and ever-changing customer requirements (Boehm & Turner 2003). Moreover, this continuous feedback is used to iteratively plan, design, code, and integrate new functionality into systems while sustaining schedule, cost, and quality (Larman 2004).

Co-located ASD suggests that team members work in one physical location (i.e. close proximity which refers to the physical distance between team members), communicate by face-to-face communications, and socially interact through informal channels (Mishra et al. 2012; Ramesh et al. 2006). Co-location remains a key belief of ASD that allows teams to communicate efficiently and effectively, and to react rapidly to changing requirements (Balijepally et al. 2009; Batra et al. 2010; Nerur et al. 2005).

In comparison, plan-driven approaches have many contrasting methods and processes (Boehm & Turner 2003). Plan-driven methods require all business needs and requirements to be defined upfront in detailed documents (Avison & Fitzgerald 2003). Unfortunately, this leaves no space for changes in requirements. That is, the development team would have to "re-invent the wheel" from the beginning if the requirements were changed (i.e. resubmitting new detailed documents and re-designing the whole system from the beginning) which will result in a high cost and long delay in delivering the product (Avison & Fitzgerald 2003). However, agile methods provide efficient and effective response to changing requirements in the hope of producing high performance software at a lower cost and shorter time (Avison & Fitzgerald 2003). Moreover, agile methods are people-oriented approaches that focus on individual skills, communication, and community rather than just focusing on processes, which increases the efficiency and agility of SD (Highsmith 2002). Hummel et al. (2013) summarise agile benefits as: shorter development time, less effort needed, better understanding of the project and individual tasks, better understanding of customer requirements, more trust among team members, better tacit knowledge sharing, better feedback, and less requirements misinterpretation. Figure 2.4 shows the differences between the plan-driven methods and agile methods regarding the costs of change and feedback cycles. The agile and plan-driven cost-of-change curves are identical. The difference is that agile methods follow techniques that keep cost at the low end of the curve (Ambler 2006).

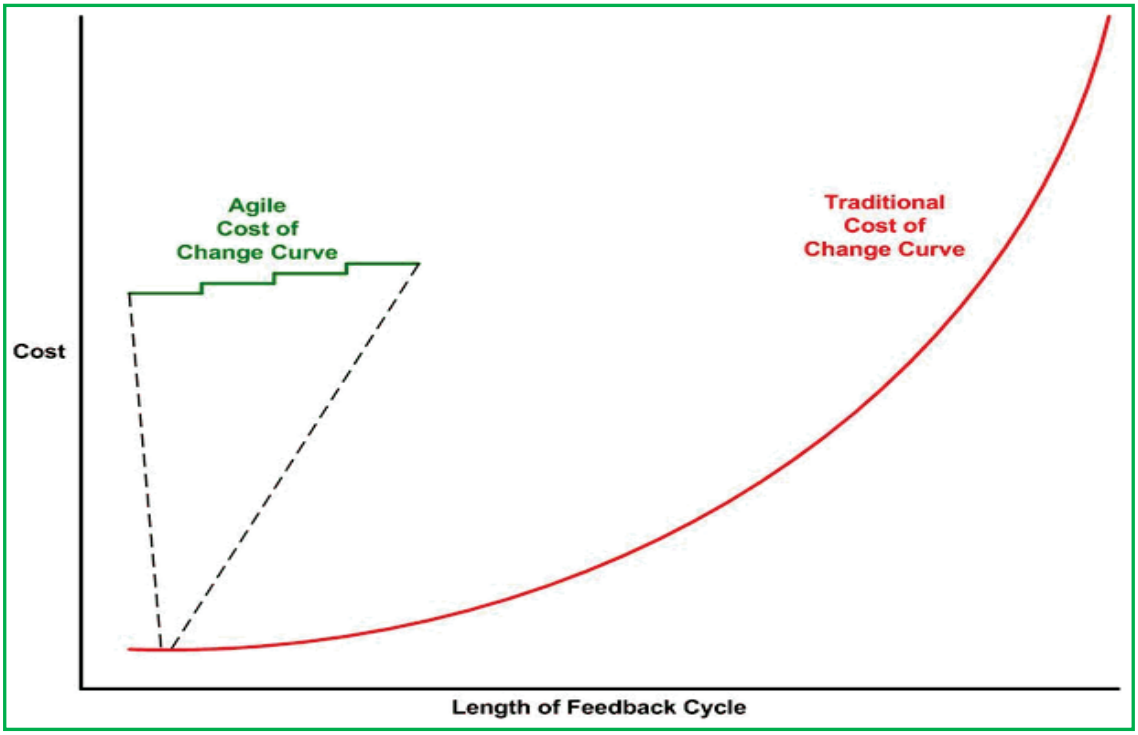


Figure 2.4. Agile and plan-driven development cost-of-change (Ambler 2006)

In their survey about agile use in 2014, Version One Inc. (Figure 2.5) found that most respondents adopted agile practices to accelerate product delivery (59%) or enhance their ability to manage changing priorities (56%). The least percentage was for better management to the distributed teams.

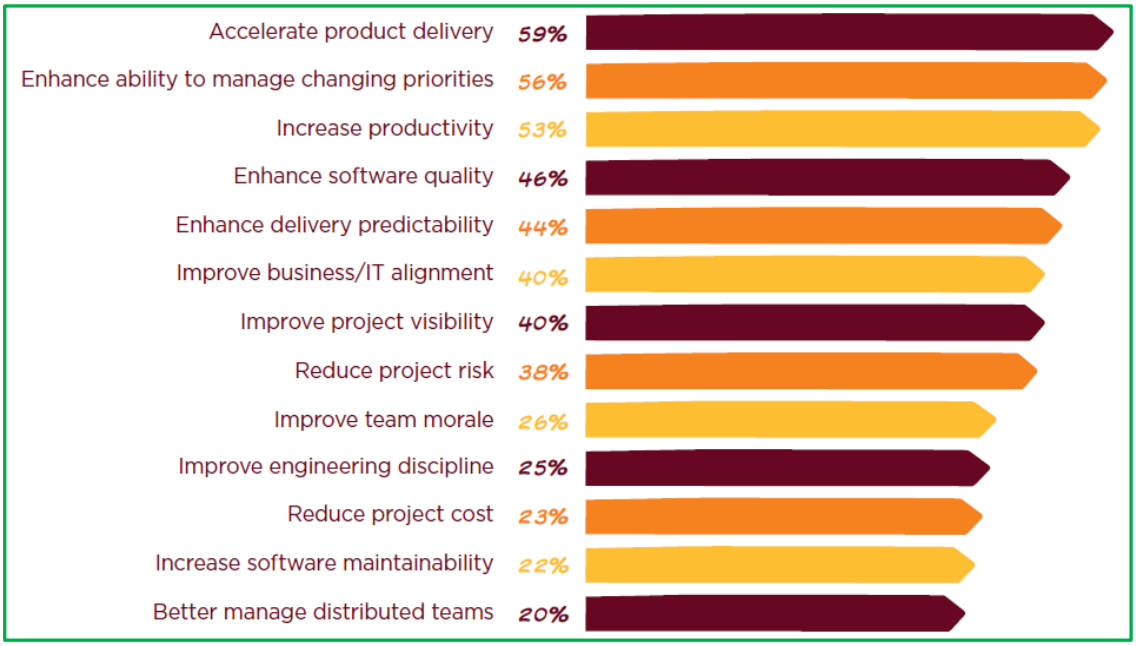


Figure 2.5. Reasons for adopting agile methods (VersionOne 2014)

2.5.1 What Is an Agility of SD?

At the heart of ASD methods is the notion of SD agility. However, agility is difficult to achieve in practice (Highsmith & Cockburn 2001). Highsmith and Cockburn (2001) define the agility as the use of light-but-sufficient rules of project behaviour and using human-and communication-oriented rules. Moreover, Agile Manifesto (2001) provides agile principles and values that qualitatively characterise agile methods. However, there is no widely-agreed, precise, and comprehensive definition of agility (Conboy & Fitzgerald 2004; Qumer & Handerson-Sellers 2008). As a result, many organisations adopt agile development approaches without a clear understanding to how agility is defined and measured, and what are the controllable factors that influence it (Conboy et al. 2010). This lack of agility understanding often results in substantial financial loss (Lee & Xia 2010).

Table 2.3 summarises the most known definitions to agility. This thesis adopts the agility definition which was introduced by Qumer and Handerson-Sellers (2008) as it represents the most comprehensive definition, so far, as:

> "Agility is a persistent behaviour or ability of a sensitive entity that exhibits flexibility to accommodate expected or unexpected changes rapidly, follows the shortest time span, uses economical, simple and quality instruments in a dynamic environment and applies updated prior knowledge and experience to learn from the internal and external environment " (Qumer & Handerson-Sellers 2006b, p. 281).

Qumer and Handerson-Sellers (2008) define agility through five attributes; flexibility (i.e. ability of an entity to adapt to changes), speed (i.e. quickness of reaching desired destination or goal), leanness (i.e. compactness and tidiness), learning (i.e. ability to improve and get knowledge), and responsiveness (i.e. reaction and sensitivity to different situations). The agility of SD increases or decreases depending on the application of these attributes.

Table 2.3: Software development agility definitions

Literature	Definition
Cockburn (2007)	Agility is using light but sufficient rules for project behaviour, human, and communication-oriented
Conboy (2009)	Agility is the continual readiness of an entity to rapidly or inherently, proactively or reactively, embrace change, while contributing to perceived customer value (economy, quality, and simplicity), through its collective components and relationships with its environment
Erickson et al. (2005)	Agility is associated with such related concepts as nimbleness, suppleness, quickness, dexterity, liveliness, or alertness; it means to strip away the heaviness in traditional software development methodologies to promote quick response to changing environments, changes in user requirements, and accelerated project deadlines
Highsmith (2002)	Agility is the ability to both create and respond to change in order to profit in a turbulent business environment; it is the ability to balance flexibility and stability
Larman (2004)	Agility is rapid and flexible response to change
Lee & Xia 2010	Agility is the software team's capability to efficiently and effectively respond to and incorporate customer requirement changes during the project life cycle
Lyytinen & Rose (2006)	Agility is defined as the ability to sense and respond swiftly to technical changes and new business opportunities; it is enacted by exploration-based learning and exploitation-based learning

2.5.2 Overview of Some of Agile Methods

In this section, a review of eight well-known agile methods is presented; Extreme Programming (XP) (Beck 2000), Scrum (Schwaber & Beedle 2002), Feature Driven Development (FDD) (Koch 2005; Palmer & Felsing 2002), Adaptive Software Development (Highsmith 2000, 2002, 2009), Dynamic Software Development Method (DSDM) (Stapleton 1997), Crystal method (Cockburn 2002, 2007), Lean Software Development (LSD) (Poppendieck & Poppendieck 2013) and Kanban Software Development (KSD) (Anderson 2010). However, only XP and Scrum methods will be discussed in detail as they are the most used methods in both co-located and distributed development (Dingsøyr et al. 2012).

XP and Scrum or the hybrid approaches (XP with Scrum) are the most used methods (Alzoubi et al. 2015; Dingsoyr et al. 2012; Hummel et al. 2013). Many practitioners use the XP method in co-located local teams and Scrum in GDAD. This thesis pays special attention to these two agile methods especially with regard to communication matters in co-located and GDAD environments. Figure 2.6 shows the most used agile methods in

practice (VersionOne 2014). These percentages represent the usage in co-located and GDAD environments.

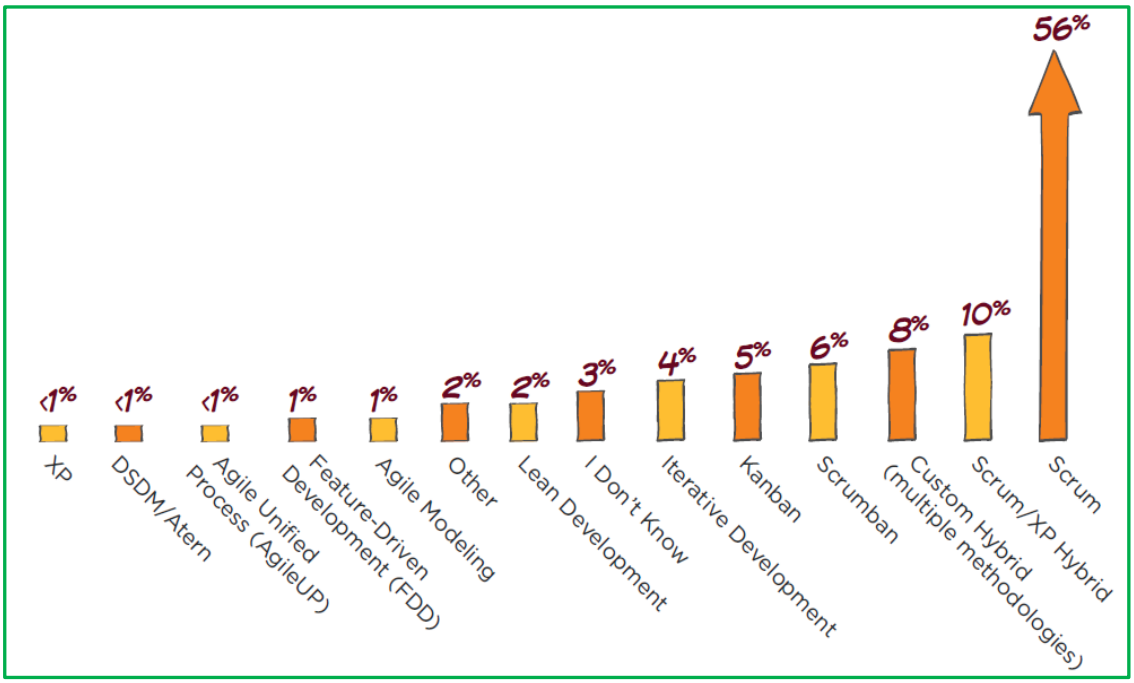


Figure 2.6. Agile methods used in practice (VersionOne 2014)

2.5.2.1 *Extreme Programming*

XP method is characterised by rapid and iterative incremental development. It improves a software project in five essential ways: communication, simplicity, feedback, respect and courage (Beck 2000). XP method can be summarised by six practices: exploration, planning, first release iterations, productionising, maintenance and death (Beck 2000). Figure 2.7 shows the different levels of XP process. XP is characterised by (Beck 2000; Plonka et al. 2011):

1. Code reviews (pair programming).
2. Testing (test driven development (TDD)).
3. Software design (relentless refactoring).
4. Simplicity (the simplest thing that could be possibly working).
5. Integration testing (continuous integration work).
6. Short iteration (the planning game).

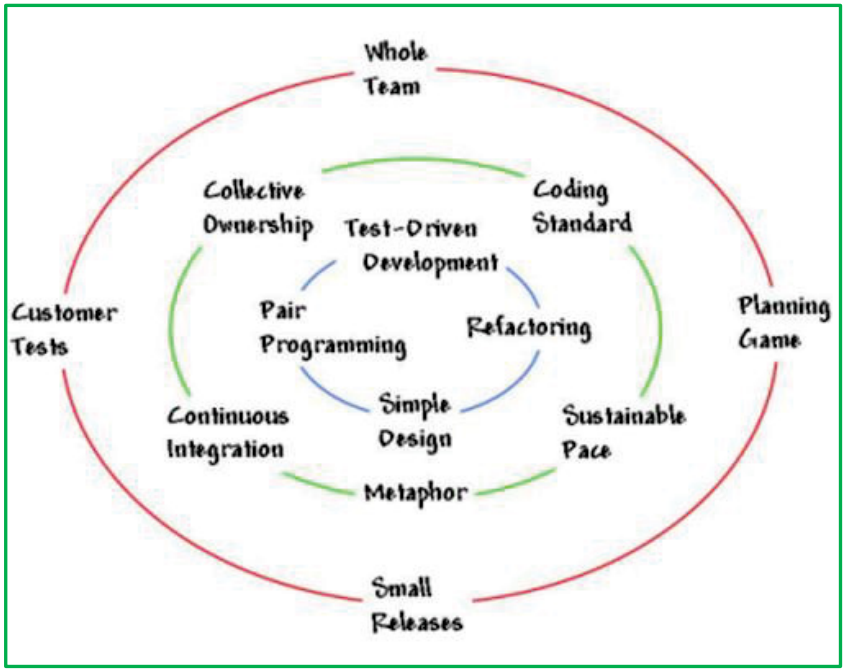


Figure 2.7. XP process (Xprogramming 2014)

Customers identify and provide the features to the development team who implements, tests, and assigns priority to the customer stories (Beck 2000). Development team implements the stories according to the customer's wish and passes any tests that the customers specify. Every cycle begins with gathering customer stories that represent the project requirements. These stories and their associated requirements are made simple but functionable. Then, the customers and development team determine the priority of the stories of which requirements will be developed in the next iteration (Beck 2000). Then, test plans are written prior to coding, followed by assigning programming pairs to specific tasks.

Upon completion, each section of code is tested according to the well-defined functional test cases. The customers must also be engaged in testing. If the acceptance test fails, the customers and development team will meet again to adjust the stories and repeat the process (Beck 2000). Upon commencing the test with expected results from the customers, the enhancements to the project are released. This is followed by completing the documentation and a final delivery of the project is made.

XP follows the fact that not everything is known at the beginning of the project. The XP planning game makes a rough plan quickly at the beginning and refines the plan as things become clearer (Beck 2000). Planning is adaptive and continuous throughout the system development process.

Small releases and the continuous design process allow for frequent review of the system by the development team and customers. XP depends on quick feedback from the customer to refine the functionality of the code that is being implemented (Beck 2000). This helps in identifying errors, supports ongoing usability tests, and builds trust between customers and the development team. The simple design approach enables frequent changes to be made to the project easily. However, XP is sometimes assumed as an untidy method that lacks discipline (Fruhling & De Vreede 2006). Some project managers view XP as a dangerous and unpredictable method because it neglects up-front planning and controlling that is necessary in large-scale projects, and because it relies primarily on tacit knowledge of the user and informal communications (i.e. the communication that happens outside the official reporting hierarchy of the team) (Herbsleb & Mockus 2003). Moreover, since XP process does not maintain specific requirements from the start of development, it is difficult to guarantee that customer's requirements will be satisfied (Fruhling & De Vreede 2006).

Refactoring is “a technique used to improve code without altering functionality, and its goal is to produce programming units with a strong internal structure. These programming units are, generally, more reusable, object-oriented, pattern-oriented, maintainable, and simpler” (Fruhling & De Vreede 2006, p. 44). Refactoring helps in reducing the complexity of the code by removing unused code and implementing patterns and in responding quickly to changing requirements (Fruhling & De Vreede 2006). However, there is no agreement on the cost of refactoring if it has to be done on an ongoing basis that may lead to extra cost and to higher project overhead (Fruhling & De Vreede 2006).

XP is an object-oriented approach that suits small and medium sized projects (i.e. team size less than 10) (Lindstrom & Jeffries 2004). XP is a simple method that discusses code style, requirement of quick feedback, applicability for co-located and distributed

teams, and seeks collaborative and cooperative action among the team and with the customer during development lifecycle (Beck 2000). High level of communication and coordination among the XP team members at all times as well as cooperation with customer are essential for efficient and effective development (Plonka et al. 2011; Sharp & Robinson 2008). In addition, the supportive culture inside the organisation and among the XP team is another successful value of XP (Qumer & Henderson-Sellers 2006c). The main impact of XP is the pair programming (Balijepally et al. 2009; Sharp & Robinson 2007), which is a continuous verbal (informal) communication process among the developers (Beck 2000). This pairing, which takes the form of low level abstraction includes frequent questions, forces more communication and enhances communication skills (Fruhling & Vreede 2006; Lindstrom & Jeffries 2004). This pair programming enhances both formal (i.e. explicit, clear communication, such as the agile requirements backlog, plans and card walls (Herbsleb & Mockus 2003)) and informal communication among team members as it was found that developers spend a quarter of their working time on communication (Layman et al. 2006).

As a result, most studies claim that improved communication is one of the main benefits of employing the XP method because it facilitates formal and informal communication (e.g., Pikkarainen et al., 2008). Table 2.4 summarises previous findings on the impact of eleven XP practices on communication.

Table 2.4: Impact of XP practices on communication (adopted from Hummel et al. 2013)

XP Practice	Impact on the Communication
Coding standards	By giving all developers the same rules for developing code, communication among team members is facilitated and simplified
Collective code ownership	When a whole team is responsible for all of the code, communication is benefited indirectly by relying on timely communication between developers when sharing code
Continuous integration	When continuously implementing new code, regular feedback on the quality of the code in the form of communication is enabled
On-Site customer / whole team	When locating the team and the customer nearby, a positive impact on communication quality is enabled. The co-located office space serves as a communication channel
Pair programming	When programming in pairs, communication skills are fostered and interactions between the programmers are improved
Planning game	Regular planning game meetings provide a frame for regular communication among the team and with the customer
Refactoring	Refactoring and simple design of the code influence communication indirectly by enabling simplicity, which is needed to communicate efficiently
Simple design	
Small releases	Small release enables more-efficient communication in meetings and provides rapid feedback mechanisms between the customer and developers
System metaphor	Metaphor serves as a communication platform with a common vocabulary that is stated to improve mutual understanding and communication quality
Unit testing (incl. Automated acceptance tests)	Automated acceptance tests are mentioned to be an effective communication medium for customers and developers to exchange business requirements, domain knowledge, and the implementation status

2.5.2.2 Scrum

Schwaber and Beedle (2002) describe Scrum method as an empirical, flexible, adaptable, productive and iterative project management approach that starts with planning and ends with review. They report that Scrum method uses the ideas of industrial process control theory for SD. According to (Schwaber & Beedle 2002), Scrum has three phases: pre-game (has two sub-phases: planning and high level design or architecture), development and post-game. Scrum key roles, artefacts, and events are summarised in Figure 2.8.

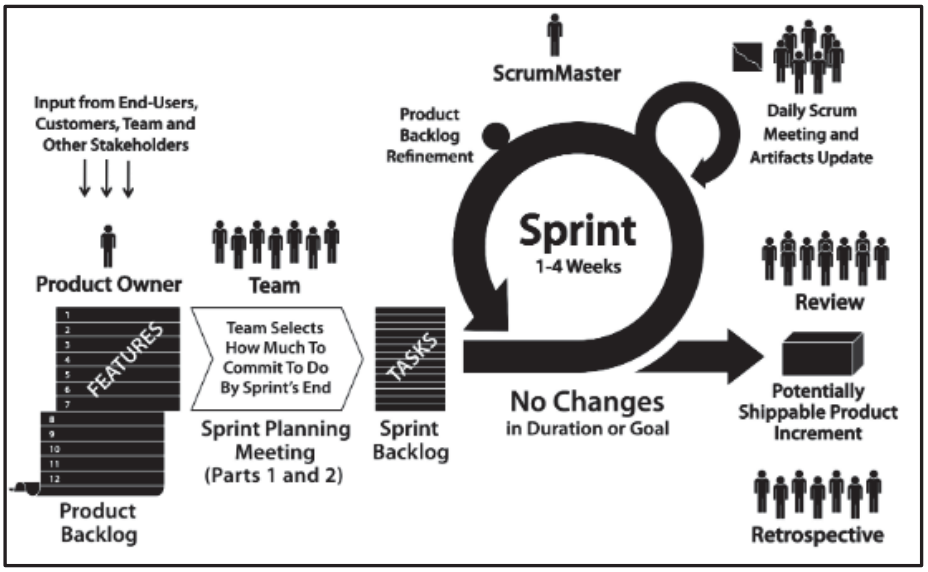


Figure 2.8. Scrum process (Sutherland & Schwaber 2011)

There are four core roles in Scrum (Schwaber & Beedle 2002; Sutherland & Schwaber 2011) to deliver a product: (1) Product Owner, (2) development team, (3) Scrum Master, and (4) stakeholders. Product Owner represents the stakeholders and is the voice of the customer, who is necessary for ensuring value to the business, and writes customer-centric items (user stories), prioritises them, and adds them to the product backlog. Development team is responsible for product increments delivery at the end of each Sprint, has 3–9 members who do the actual work (analyse, design, develop, test, technical communication, document, etc.), and it is self-organised. Scrum Master is accountable for removing impediments facing the team to deliver the Sprint as scheduled, ensures that the Scrum method is used as it should be and enforces rules, keeps team focused on the current tasks, reinforces the dual perspectives, and excludes additional responsibilities from members. Product Backlog is an ordered list of the project's requirements. It consists of whatever needs to be done or features (e.g., bug fixes, non-functional requirements, and so on) in order to successfully deliver a working product. The Product Backlog is enough for a first Sprint to start with, and then grows and changes as more is learned about the product and its customers. Sprint Backlog is the list of tasks that need to be addressed during the next Sprint. Scrum iteratively overlaps the requirements, design, code, and test processes (Schwaber & Beedle 2002).

Like all other ASD methods, Scrum process starts by adding “user stories” (customer requirements) to the Product Backlog. The key activities during this stage of the Scrum

process include high-level design and architecture for the product (Schwaber & Beedle 2002). This is followed by Sprint planning where the customer and developers jointly analyse the Sprint Backlog and prioritise the features and functionality to be incorporated into the Sprint. Then, the developers together estimate the effort needed for planning, designing, and delivering the new features in the product within the Sprint's time period (Schwaber & Beedle 2002). During the Sprint, the team designs, codes, integrates, and tests the system. As the team progresses, additional items may be pulled up from the Product Backlog. The daily Scrum meeting is held by all team members to discuss the progress in the work, the challenges, and a possible re-estimating regarding further work. During the scrum session, usually three questions are posed; what did you do yesterday? What are you going to do today? And what are you going to do tomorrow? This repeated evolution of designing, coding, integrating and testing occurs until the Sprint is completed. After the commencement of a Sprint, no changes to requirements can be introduced. Upon completion of a Sprint, a review of theto Sprint is conducted, a retrospective is performed, and some important documentation is completed. The team discusses the successes, challenges, and other performance elements for the future during the retrospective. The output of this iteration is a workable (shippable) product (system or software) with full functions.

It is crucial for the team to clearly and uniquely approach each activity and iteration for entirely successful integrated and completed development process. For example, it is very important to make sure that the correct and understandable messages are sent and received between team members particularly in GDAD environment where many communication media are used. This ensures the high performance of the product. Since customer requirements change continuously, communication between team members is critical for development success.

Scrum, as an iterative and incremental object-oriented approach, suits small, medium and large sized projects. Scrum team size is less than 10 people but it can be expanded to multiple teams (distributed teams) on the same project (Sutherland & Schwaber 2011). Unlike the XP method, Scrum does not explicitly discuss code style, requirement of quick feedback, applicability for co-located and distributed teams (Qumer & Henderson-Sellers 2008).

Scrum is a collaborative and communication-oriented SD method. As such, key principles of Scrum are "keep everything visible and engage everyone in identifying and alleviating obstacles" (Schwaber & Beedle 2002). The Scrum sessions are important communication mechanisms to synchronise all team members and are traditionally done in face-to-face meetings within a co-located team. Furthermore, effective collaboration and coordination among all team members is compulsory during team meetings. Outside the daily Scrum meetings, team members collaboratively engage in communications about the system (design, coding, integration, test, and so on). As for XP method, documentation in Scrum method is very limited assuming that the communication among team members during the Scrum sessions must be comprehensive (Schwaber & Beedle 2002). Previous studies have mentioned that the Scrum process enhances members to work together closely and Scrum Masters enforce team members to get involved in the communication during the meetings (Hummel et al. 2013). This increases the frequency and the quality of communication among team members.

Table 2.5: Impact of Scrum practices on communication (adopted from Hummel et al. 2013)

Scrum Practice	Impact on the Communication
Daily stand-up meetings	The daily meetings are judged to be one of the most important mechanisms for communication because they ensure that team members talk regularly with each other. Daily meetings are reported to have positive effects on the communication frequency among the whole team and one-on-one communication, after the meetings, is also encouraged
Iteration planning meetings	These meetings, which are scheduled once during iteration, are used for communication about features, requirements, and lessons learnt. Both the development team and the customer representative are engaged in this regular communication
Sprint review & retrospective meetings	
Scrum-of-Scrums	Regular meetings of representatives of different teams enable communication and coordination between several teams. Negative effects of those meetings include the possibility of miscommunication and misunderstanding because of the reliance on a single person that represents the team

As a result, most studies have claimed that Scrum enables a higher communication frequency and communication quality because it provides a frame for communication that reminds team members to interact closely and regularly (e.g., Paasivaara et al.

2008). Table 2.5 summarises the impact of the previous findings of individual Scrum practices on communication (Hummel et al. 2013).

2.5.2.3 Other Agile Methods

In this section, a brief review is presented of some of the other agile methods (i.e. FDD, Adaptive SD, DSDM, Crystal method, LSD, and KSD).

FDD is an object-oriented approach that is characterised by design and planning phases, which differentiates it from other development approaches (Koch 2005). FDD also focuses on iterative and construction design (Palmer & Felsing 2002). FDD provides developers with guidelines, tasks, supportive techniques. FDD also provides the following processes: develop an overall model, build a feature list, plan by feature, design by feature, and build by feature (Palmer & Felsing 2002). FDD suits small, medium and large sized projects such that there is no limit on team size (Qumer & Henderson-Sellers 2008). Like Scrum method, FDD does not explicitly discuss code style and requirement of quick feedback, and it is applicable for co-located and distributed teams (Qumer & Henderson-Sellers 2008).

Adaptive SD was developed by Highsmith (2000) as a framework that provides guidelines, concepts and practices in order to encourage agile development. It is an object-oriented approach with increment, iterative and constant prototyping development that helps in complex systems. This method has three phases: speculate, collaborate and learn (Highsmith 2000). It can be used in GDAD as it addresses cultural and team skills issues, and also provides techniques to deal with them (Qumer & Henderson-Sellers 2008). Hence, adaptive SD can be used in co-located or distributed teams at different geographical locations, large complex systems where there is no limit to team size. However, as Scrum method, it does not explicitly address code style and requirement of quick feedback, and it is applicable for co-located and distributed teams (Qumer & Henderson-Sellers 2008).

DSDM was started by Stapleton (1997). It provides a framework to support iterative, rapid, and collaborative development (Abrahamsson et al. 2002). Two phases are

distinguished in DSDM: basic principle of the resources and timeframe are adjusted, and then the goals and the required functionality are adjusted (Qumer & Henderson-Sellers 2008). However, DSDM consortium addresses seven phases: Pre-Project, feasibility study, business study, functional model iteration, design and build iteration, implementation and post-project (DSDM 2013). The main features of DSDM are (DSDM 2013): (1) user involvement, (2) iterative and incremental method, (3) increased delivery frequency, (4) integrated test at each phase, and (5) the acceptance of product depends directly on fulfilling requirements. DSDM is an object/component-oriented approach that suits iterative rapid SD. So, it can be used in small, medium and large projects. Also, DSDM project can be extended to have multiple teams with 2 members (minimum) to 6 members (maximum) for the same team (Qumer & Henderson-Sellers 2008). DSDM addresses collaborative and cooperative business culture, which means that the relationship between the development company and the customer is open. As for Scrum and Adaptive SD, DSDM does not address code style and technology environment, but it addresses the physical environment (Qumer & Henderson-Sellers 2008).

Crystal methodologies are incremental developments such that one increment may take several iterations to complete and more than one increment may take place in parallel (Cockburn 2007). The common Crystal methodologies lifecycle can be: envisioning, proposal, sales, setup, requirements, design and code, test, deploy, train, and alter (Cockburn 2002). Crystal methodologies are represented by different coloured teams such as Crystal clear that has one team (Abrahamsson et al. 2002), whereas Crystal orange team may be split into several cross-functional groups (Cockburn 2002) by using the holistic diversity strategy (Cockburn 2007). Like the XP approach, Crystal methodologies are people-oriented approaches such that developers can monitor the development processes continuously during progress lifecycle (Cockburn 2007). There are some guidelines for Crystal methodologies such as policy standards, tools, work products, and local matters (Cockburn 2002). Qumer and Henderson-Sellers (2008) argue that Crystal methodologies are object-oriented and provide different methodologies such as Crystal clear and Crystal orange. These methodologies enable Crystal projects to be extended from small (6 members maximum for one team) to large projects (multiple teams are included). However, Crystal doesn't support GDAD, and

DSDM Crystal does not explicitly specify code style, technology and culture environment (Qumer & Henderson-Sellers 2008).

LSD principles have been adapted from Toyota manufacturing systems (Poppendieck & Poppendieck 2013). Later on, these principles were used in the IT industry and then ASD (Poppendieck & Poppendieck 2013). According to Poppendieck & Poppendieck (2013), LSD has seven principles:

1. focus on customer: the best way to solve the problem is to focus on the customer problems rather than focusing on building or developing the product. Customer's focus results in "loving" your product rather than just be convenient about it,
2. energise workers: software developers need to be energised and inspired to do their tasks, provided with challenge and feedback, and provided with suitable environment,
3. eliminate waste: anything that does not add value to the system or customer is considered waste such as unnecessary code, unclear requirements, and delay in software process or communication process,
4. learn first: SD organisations should focus on quick learning and rapid responding to the changes. This requires adopting a knowledge-based development environment that supports multi-solutions and synchronised team members,
5. deliver fast: the high competitive environment needs fast product delivery with high quality, meeting customer's needs and competitive cost. The focus then needs to be on flow efficiency rather than resource efficiency, and on workflow management rather than task-based schedules,
6. build quality: developers need to find and fix the defects when they occur so customers have to be involved and informed always, and
7. keep getting better: to keep competitive, SD organisations need to be ready for change in the same rhythm as the world does. Hypothesis, experiments, documentation, and the best alternative implementations are all essential for better development.

KSD is a knowledge management method that emphasises delivering the product on time (Anderson 2010). It is an incremental approach that suits small changes, uses work-in-progress mechanism to detect and fix the defects, and stimulate collaboration to improve the product (Anderson 2010). According to Anderson (2010), there are five practices that distinguish KSD from other agile methods: visualise the work flow, limit work-in-progress mechanism, manage work flow, make policies explicit, and implement feedback process. KSD has four principles (Anderson 2010; Scotland 2013): (1) start with what you know: KSD does not have specific steps or procedures, rather it starts with what the developer knows about the system and uses incremental evolutionary changes to the system, (2) agree to pursue the incremental evolutionary changes: team or organisation need to agree that the way of system improvement is through continuous small incremental evolutionary changes, (3) respect the current rules, responsibilities, and process, and (4) be under leadership at all levels.

2.6 Distributed Traditional Software Development

Globalisation and the high improvement in communication technologies have created a new type of business market competition and development (Herbsleb 2007). This encourages SD organisations to adopt DSD that has some advantages over the co-located local development such as low cost, proximity to the market, quick formation of virtual teams, "round the clock" development, high quality and flexibility (Carmel et al. 2010; Herbsleb & Moitra 2001). DSD can be defined as using distributed teams in different geographical locations and different time-zones, either in the same country or in different countries, such that teams working together accomplish the project goals (Layman et al. 2006). Outsourcing and distributed teams within the same organisation that are located in different locations in the same country or in different countries both represent examples of DSD (Layman et al. 2006).

The results of using DSD are multisite SD, multicultural teams, and globally distributed undertakings. As a result, there are needs for high level arrangement, management and communication technologies (Paasivaara & Lassenius 2003). However, the potential benefits of DSD and the fact that the only way to gather all talented developers around

the world is through geographically dispersed teams are behind adopting DSD (Herbsleb & Moitra 2001).

2.7 Geographically Distributed Agile Development

After the overview about agile methods and their impact on communication within co-located teams, and DSD discussed in previous sections, here we move closer to the focus of this research; the GDAD. The success of agile development in co-located small and medium size teams motivates large software organisations to scale and adopt agile approaches in the GDAD environment (Abrahamsson et al. 2009; Dingsøyr et al. 2014). Over some years, the body of knowledge on GDAD has constantly grown (Dingsøyr et al. 2012) to overcome the lack of empirical and rigorous studies (Agerfalk et al. 2009; Dingsøyr et al. 2014; Laanti 2014). However, there is a number of challenges of GDAD (e.g., da Silva et al. 2010; Hossain et al. 2009b; Shrivastava & Rathod 2015).

ASD is considered to be a management fad (Gill et al. 2012). That is, using the existing good practices and giving them new terminology (Agerfalk et al. 2009). Agile methods suit more developmental rather than sequential projects, adopt mixed elements of traditional and agile developments, and suit small projects where team's members are co-located (Agerfalk et al. 2009). This is true from the angle that agile methods focus on the frequency of informal communication between team members and with customers. Therefore, to get full benefit of ASD, it is easier and more effective to be in a small co-located team (Batra et al. 2011). However, many authors reported that ASD can be scaled in distributed and multi-teams using the current technologies and tools such as Wiki, instant messaging and cloud computing (Gill et al. 2012; Layman et al. 2006; Paasivaara et al. 2008; Paasivaara et al. 2013). In the following argument, we discuss and summarise the previous literature regarding the doablity of GDAD.

ASD has its perceived shortfalls such as the lack of adequate scaling during large projects or in distributed environments, as opponents of agile contend (Boehm & Turner 2003; Gill et al. 2012). Although it is widely accepted that ASD enhances agility, the supporting evidence is generally restricted to small co-located projects (Boehm & Turner 2003; Batra et al. 2011). Fruhling and De Vreede (2006) states that:

> "Agile methodologies are intended for smaller, less-complex information system projects with less than ten team members. Large-scale systems that require more than ten team members are better served with plan-driven methodologies. Agile methods can be difficult to scale up to large projects because of a lack of sufficient architecture planning and over-focusing on an early result. Moreover, in large, complex systems with less-experienced developers using the plan-driven approach, refactoring is expensive. Agile approaches with skilled developers and small, less-complex systems benefit from refactoring being essentially free" (Fruhling & De Vreede 2006, p. 42).

Chow and Cao (2008) examined critical success factors in 109 agile projects. They found that most of the project teams (80%) had less than 20 members. The Scrum methodology is recommended to projects of teams that do not have more than six members (Schwaber 2004). Beck (2005) states that XP project probably couldn't be run with a hundred programmers; not fifty, nor twenty, and ten members is definitely doable. Moreover, Sarker and Sarker (2009) suggest that adoption of agile methods is one of the several potential contributors to DSD team agility, and that agile methods need suitable adaptation for effective adoption/use in GDAD environment.

The majority of plan-driven methods in distributed environment are structured, and rely on mechanisms of indirect communication such as project plans, specification documents, or models to support and control coordination, communication, and knowledge sharing among team members (Boehm & Turner 2003). In dynamic and rapidly changing environments, however, these formal mechanisms impede quick reactions to changes (Cockburn 2007; Hummel et al. 2015), which make this environment appropriate for ASD (Batra et al. 2011). This tension in GDAD poses conflicting demands between alignment and adaptability in the SD process (Ramesh et al. 2012). Whereas DSD benefits from formalised processes recommended by plan-driven (process-oriented) development approaches (Thomas & Bostrom 2010; Carmel et al. 2010), agile development is a more people-oriented approach (Qumer & Henderson-Sellers 2006a). In fact, agile development refers to individual autonomy (i.e. the freedom of agile team members to act of their own volition) as one of its values

(Maruping et al. 2009). Furthermore, agile project management counts on leadership and collaboration instead of command and control (Larman 2004).

In addition, a GDAD project cannot be up-front planned very far into the future because the business environment changes, customers may change their requirements or adjust the system once they have seen an example, and estimating how long it will take to meet requirements is only an estimate (Fruhling & De Vreede 2006, p. 62) so "when we build plans, we need to make sure that our plans are flexible and ready to adapt to changes in the business and technology" (Martin 2003, p. 11). That is, "not spending a lot of time on designing and developing the architecture for current and future requirements; rather, only the architecture infrastructure the project needed for the current requirements is designed and developed, which keeps the system simple, increases developments speed, and reduces overhead" (Fruhling & De Vreede 2006, p. 64).

Furthermore, plan-driven distributed development depends heavily on documentation (Martin 2003). Even if the document content is not up-to-date, it can be relevant and can serve as an important tool for communication and should always serve a purpose (Fruhling & De Vreede 2006). Agile development does not prefer documentation; rather it prefers the continuous communication between team members and with the customer (i.e. counts on tacit knowledge) (Martin 2003). However, "agile developers are more likely to agree that documentation should be brief and to the point" (Fruhling & De Vreede 2006, p. 61). In fact, "the importance of system documentation for software engineering is being debated in the literature" (Fruhling & De Vreede 2006, p. 61).

Although the disciplined plan-driven methods have been popular, many experts (e.g., Boehm & Turner 2003) question the effectiveness of the bureaucratic nature of plan-driven approaches in today's dynamic business world. Many practitioners (e.g., Highsmith 2001) have argued that extreme planning, documentation, and control may run counter to the uncertainty-marked nature of the current software project (Batra et al. 2011). Therefore, it is important to evaluate the feasibility of extending agile methods to large and distributed environments to come up with a more balanced approach (Batra et al. 2011; Fitzgerald et al. 2006).

It is widely accepted that agile development enhances agility although the supporting evidence is generally restricted to small projects (Boehm & Turner 2003; Batra, VanderMeer & Dutta 2011). However, small projects account for only 17% of programming code (Beck & Boehm 2003). Even open-source development can be large and significant enough to disrupt commercial incumbents (Batra et al. 2011). Moreover, due to frequently changing requirements and severe time-to-market constraints (Herbsleb 2007), SD organisations which have been using plan-driven methods have progressively recognised the need to incorporate flexibility in development through employing agile approaches (Batra et al. 2011).

Nevertheless, agile values and principles can provide a starting point to examine agility for large, dynamic projects (Batra et al. 2011). A few studies (e.g., Batra et al. 2010; Barlow et al. 2011; Cao et al. 2009; Mishra et al. 2012; Ramesh et al. 2012; Sarker & Sarker 2009) have examined attempts to introduce agile methods in GDAD or large development organisations. Some authors (e.g., Nerur & Balijepally 2007) have reported that there is some empirical evidence that agile development organisations are attempting to employ both agile and organisational approaches. The extant literature offers limited guidance of GDAD teams and how project leaders should manage the balance between structure and autonomy (Maruping et al. 2009). Therefore, GDAD will be more effective when paired with the exercise of outcome (control), rather than self-control (Maruping et al. 2009).

In general, there are no clear guidelines that specify which agile principles are applicable for large projects (Abrahamsson et al. 2009; Batra et al. 2011). In fact, there is some uncertainty about the efficacy of agile methods in GDAD environment (Batra et al. 2011). Few authors claim that GDAD should use some sort of plan-driven practices to success (e.g., Maruping et al. 2009; Ramesh et al. 2012). However, this claim is still under investigation, at least from the academic point of view since most successful GDAD stories have been reported from industrial experience (Batra et al. 2011; Hummel et al. 2015).

To a large extent, GDAD challenges invariably consist of communication deficits, as well as the competition between cultures and organisational norms, which routinely adds to software project failures (Abrahamsson et al. 2009; Agerfalk et al. 2009; Martini et al. 2013). The holistic insight of agile is one sided (Gill et al. 2012; Mishra et al. 2012; Modi et al. 2013); agile relies largely on communication between team members and with customers, and within large and distributed teams, this can prove to be challenging (Batra 2009; Dingsøyr et al. 2014; Mishra et al. 2012; Ramesh et al. 2006; Sarker & Sarker 2009). Due to the lack of face-to-face communication in distributed environment, many issues face which face GDAD are similar to distributed plan-driven development (Agerfalk et al. 2009), and the communication issue is the most noticeable challenge (Herbsleb & Mockus 2003, Hinds & Mortensen 2005, Thomas & Bostrom 2010). Consequently, the communication frequency is dramatically decreased, the inability to maintain interpersonal ties increases, and the difficulty in finding appropriate information increases (Layman et al. 2006). This is due to a number of GDAD communication challenges such as distance differences, time-zone differences, culture differences, project domain, and process management (Agerfalk et al. 2009; Hossain et al. 2009a; Shrivastava & Rathod 2015).

GDAD communication is criticised for its dependence on technology (Korkala & Abrahamsson 2007). Moreover, agile development requirements are managed by effective informal communication and promises less documentation, which is hard to be achieved in GDAD (Korkala & Abrahamsson 2007). Figure 2.9 shows the effectiveness of different communication means (channels). Communication effectiveness (e.g., “richness") is based on the means capacity to facilitate insight, shared meaning, and rapid understanding (Daft et al. 1987). The figure also shows the differences between co-located agile teams and GDAD communication effectiveness and tools used.

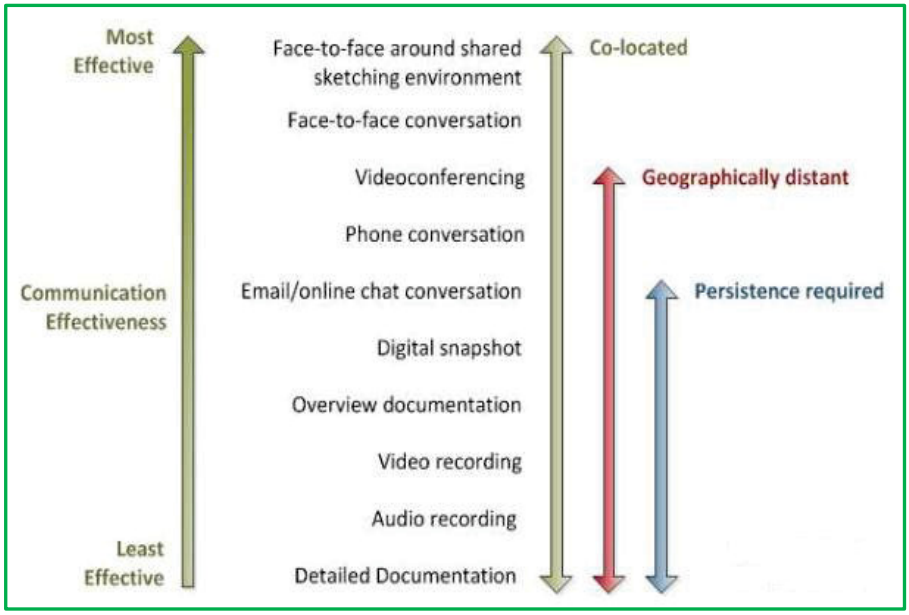


Figure 2.9. Effectiveness of communication means (Ambler 2015)

Many studies have discussed agile communication in large agile organisations within the same time-zone and geographical context (e.g., Ali Babar et al. 2009; Kuusinen et al. 2012). Kuusinen et al. (2012) reported that process management and communication are the main problems facing large teams. They argued that communication problems could be due to the reduced opportunity for team members to meet face-to-face, a lack of synchronisation among team members and a lack of frequent feedback between users, designers and architecture teams (Kuusinen et al. 2012). Ineffective communication may lead to many issues, such as a lack of collaboration and coordination between different teams and a lack of understanding of customer requirements (Kuusinen et al. 2012). Ali Babar et al. (2009) reported that cross-team communication is the greatest problem facing large teams. They argued that the ideal way to overcome this issue is by reducing cross-team communications (Ali Babar et al. 2009).

Martini et al. (2013) reported five challenging communication categories: (1) architecture (i.e., unnecessary flow or misunderstanding of communication due to the definition of the system and software structures), (2) technology (i.e., the differences in the tools and programming languages used by GDAD teams), (3) processes (i.e., the identification of who is responsible for delivering what), (4) organisation (i.e., the structure of task allocation, coordination and supervision), and (5) people (i.e., personal

or group attitude, mind-set or knowledge). The authors found that the architecture category was very significant for system agility or reuse. The authors suggested using reference architecture, achieving agreement on the high-level system requirements at the beginning of a project, and including reusability as an important quality. Process represents a boundary for development in general. The authors suggested providing developers with frequent interfaces for strategic aspects, coordinating these aspects with the overall development strategy, the local process, and adjusting the team attitude to match the overall strategy. Team distribution was found to slow the development. The authors suggested promoting frequent onsite meetings and using social media tools to support and increase the frequency of GDAD communication. Organisational differences between GDAD teams were found to negatively affect communication. The authors suggested avoiding organisational dependencies caused by the architecture. They also suggested relocating team members working on different projects to promote knowledge sharing and build relationships among members of different teams.

Sutherland et al. (2009) and Sutherland et al. (2007) reported that issues related to team culture and company culture in GDAD environemnt have a negative impact on communication between GDAD teams. The authors claimed that using Scrum practices (e.g., Scrum team, Product Owner, and Scrum Master) decrease the effect of GDAD challenges and increase productivity (Sutherland et al. 2009; Sutherland et al. 2007). In addition, using scalable Scrum (e.g., Scrum of Scrums) implementations with minimal tooling has been suggested to support real-time information sharing (Sutherland et al. 2009; Sutherland et al. 2007). Some strategies have been suggested to enhance communication between distributed Scrum teams (Sutherland et al. 2009), such as Scrum meetings that facilitate all necessary communication between distributed teams, separate meeting rooms that are equipped with communication tools such as “digital burn down charts” and video conferencing with audio equipment, and local meetings in addition to the cross-team meetings. The aim of these strategies is to ensure that requirements are clear to developers.

Dorairaj et al. (2011) reported four GDAD communication challenges: different time-zones, communication tools, language barriers, and teamwork tasks. Firstly, having team members located in different time-zones limits the communication opportunities

and makes it difficult for GDAD teams to organise group meetings outside the local standard working hours. To address the different time-zone challenge, the authors suggested minimising time-zone differences between GDAD teams such that members with only small time-zone differences are included in a given GDAD team (Dorairaj et al. 2011). Secondly, the available communication tools can be used to support communication among GDAD teams. However, communication tools may negatively impact team communication if these tools are not tailored and systematically deployed to fit the purpose of GDAD (Gill & Bunker 2013). Thirdly, the use of a variety of languages may limit communication between GDAD teams, even if developers are proficient in common foreign business communication languages, such as English. It has been noted that speaking slowly, informing team members about the topics of discussion in the meeting and addressing the language issue as early as possible in the project are strategies that may reduce the impact of language barriers (Dorairaj et al. 2011). Finally, teamwork requires all members to communicate and understand each other, even if some members do not wish to communicate (Dorairaj et al. 2011). Dorairaj et al. (2011) recommended enhancing trust and facilitating effective formal communication (e.g., via an inception meeting at the beginning of the project, weekly or daily meetings with other distributed members, and meetings with customers) and informal communication among team members.

Layman et al. (2006) studied distributed XP teams' communication in the USA and the Czech Republic. They studied the effect of temporal distance (time-zone), geographical distance, customer communication, technology (communication tools), and different languages on the communication among GDAD teams. The authors recommended some guidelines using XP practices in a GDAD environment, such as:

- encouraging developers to work closely with project management teams on a daily basis,
- assigning an individual to play the role of the customer up front,
- using the available asynchronous communication tools as a substitution for face-to-face meetings, and
- using the available project management tools to record and track the daily development progress.

XP method was reported to have a positive impact on the GDAD communication (Layman et al. 2006; Lee, Judge & McCrickard 2011; Sharp et al. 2012; Schümmer & Lukosch 2008). The most beneficial aspect of XP is increasing the direct communication among team members because of less documentation used in development (Barlow et al. 2011; Cao et al. 2013; Fruhling & de Vreede, 2006; Kollmann et al. 2009; Layman et al. 2004; Ramesh et al. 2010). Schummer and Lukosch (2008) reported that the biggest disadvantage of geographically distributed XP, comparing to the co-located XP development, is the ineffective communication between distributed teams. If communication among teams is hindered, they will be less aware of each other and the coordination and the collaboration among them will be less, which will negatively impact the performance of the project (product) (e.g., Lee et al. 2011; Sharp & Robinson 2007; Schummer & Lukosch 2008). Layman et al. (2006) found that using XP methodology for large distributed teams resulted in more than average team productivity, which is bit higher than without using XP method. Moreover, Scrum method was reported to be effective in GDAD environment (Hossain et al. 2011; Paasivaara et al. 2013; Sutherland 2009). Moreover, some authors have reported the importance of daily Scrum as a means of GDAD communication (e.g., Sutherland et al. 2009; Sutherland & Schwaber 2011; Zieris & Salinger 2013). Scrum practices enables higher communication efficiency and effectiveness in GDAD as it provides a frame for communication that ensure that the team members interact closely and regularly (Alzoubi et al. 2015; Paasivaara et al. 2013, 2014; Sutherland et al. 2009). As a result, most studies claim that improved communication is one of the main benefits of employing XP and Scrum practices because they facilitate formal and informal communication (e.g., Alzoubi et al. 2015; Pikkarainen et al. 2008). However, other studies have reported that no differences had been noticed after using XP in GDAD environment on the way and how often the distributed team members communicated between each other, with the customer, or with customer representatives (Heiberg et al. 2003; Hulkko & Abrahamsson 2005; Svensson & Höst 2005; Vanhanen & Lassenius 2005). Moreover, other studies (e.g., Vallon et al. 2013) recommended that Scrum method should not be scaled to more than one site.

Informal communication is important in GDAD and it should be known how communication should be structured so that team members have the information they need to get their work done without incurring a heavy overhead (Cataldo & Ehrlich 2011). Purely informal communication may be difficult or even not doable in distributed projects (Batra 2009; Batra et al. 2011). Direct communication may, therefore, not always be the best choice and may also become an inhibitor to communication effectiveness (Vidgen & Wang 2009). However, other studies reported that informal communication can be problematic in complex GDAD projects compared to simple co-located agile projects (Korkala & Abrahamsson 2007). Depending on the use of excessive informal communication represents a challenge, especially if the project members have weak communication skills or if inadequate technology hinders communication with external parties or teams (Hummel et al. 2013).

Traditionally, to mitigate the effect of communication challenges in distributed environment, some practices of plan-driven are employed such as using formal forms of communication in addition to the informal forms (Hinds & Mortensen 2005, Thomas & Bostrom 2010). These formal forms include some artefacts such as architecture description, process specifications, and project plans (Barlow et al. 2011; Hinds & Mortensen 2005; Ramesh et al. 2012). Other studies suggested applying control through well-defined and formal processes (e.g., detailed architecture and heavy documentation), and formal coordination tools and techniques such as workflow management systems and configuration management tools (e.g., Bostrom 1989; Carlile 2004; Carmel et al. 2010; Kirsch 1996).

Some of the previous studies support using some of the plan-driven principles and emphasise the role of indirect communication (e.g., not face-to-face) in the GDAD such that the architectural and requirements specifications should be documented at a high level of abstraction (e.g., Selic 2009; Stettina et al. 2012), customer feedback should be documented (Hoda et al., 2010), customer requirements and major decisions need to be documented (Batra et al. 2010; Stettina et al. 2012), project dictionary (e.g., business terms, their meanings, and contexts of use) should be documented, and knowledge that is relevant for all project members should be documented (Hoda et al. 2010; Stettina et al. 2012). However, documentation at the code level is judged to be not useful (Selic

2009). Inaddition, the use of plan-driven principles opposes agile development that strongly emphasises constant communication among team members and with customers, particularly through face-to-face interaction over written specifications (Boehm & Turner 2003; Highsmith & Cockburn 2001), counts on self-organising teams (Boehm & Turner 2003) which requires trust among team members, team cohesiveness, and strong interpersonal relationships (Larman 2004), and tries to minimise the overhead associated with formal procedures and to increase process agility by emphasising interpersonal coordination (Boehm & Turner 2003).

Nevertheless, none of the previous successful GDAD projects that have been reported in literature used pure agile development (Alzoubi et al. 2016). In a nut shell, in most successful GDAD projects nowadays, "either pure agility or pure plan-driven alone will not meet the needs. Combining agile and plan-driven methods is necessary" (Fruhling & De Vreede 2006, p. 64). Many agree that no methodology is adopted 100 percent; there is always some adjustment (Batra et al. 2011; Fruhling & De Vreede 2006; Ramesh et al. 2012).

2.8 Literature Gaps

Based on this extensive literature overview, it can be concluded that there is an increasing interest in using GDAD, and industry experience suggests that agile practices (e.g., XP and Scrum) promote communication, collaboration and frequent product delivery, but significant challenges need to be mitigated. The above review is summarised into three major aspects.

First, due to the limited in-depth empirical studies, current literature provides limited insight into the GDAD communication issue. Little is known about how efficient and effective GDAD communication is achieved in practice, what techniques can be used to enhance GDAD communication, how GDAD communication challenges can be mitigated, and how agile practices enhance GDAD communication (Agerfalk et al. 2009; Dingsøyr et al. 2012; Gill & Bunker 2011; Herbsleb 2007; Hummel et al. 2013; Korkala & Maurer 2014). In addition, the GDAD communication challenges and techniques seem to be scattered "everywhere" and from "everywhere". Therefore, there

is a research gap in the theory and practice of using GDAD. To begin to formalise a response to this gap, the following research questions needs to be answered through surveying the empirical literature of GDAD. The answer to these questions; however, is discussed in the following Chapter (Ch. 3).

RQ1: What are the challenges or factors that limit DAD communication?

RQ2: Which strategies, techniques or practices have been used to overcome these challenges and enhance GDAD communication?

Second, despite the growing interest in adopting GDAD, most of the suggestions to improve GDAD communication tools, techniques, and practices have come from experienced practitioners (Abrahamsson et al. 2009; Dingsøyr et al. 2012). To obtain a deeper understanding of GDAD communication, previous authors have recommended that the research design needs to fit the current state of theory and research (e.g., Dingsøyr et al. 2014; Dybå & Dingsøyr 2008). Therefore, there is another research gap in the theory. This study will try to fill up this gap by theorising the GDAD communication using Common Ground theory, which will be discussed in detail in Ch. 4.

Finally, as the above argument shows, there are few studies that investigated GDAD challenges in general, and a few studies that investigated or reported GDAD communication, specifically. Despite that, most of these studies have reported that pure agile methods in distributed environment may be not doable (e.g., Batra et al. 2011; Ramesh et al. 2012). This fact calls for two contradictory empirical research types; (1) empirical studies that only used agile principles in GDAD, if any, and (2) empirical studies that employed some of distributed plan-driven methods principles in GDAD. This study will investigate the second type (i.e. plan-driven principles in GDAD) through employing agile EA as a means of GDAD communication. This will be discussed in Ch. 3 and Ch. 4.

2.9 Chapter Summary

This chapter has reviewed the evolvement of SD that starts with traditional plan-driven methods to agile EA driven GDAD method. DSD development relies on formal forms of communication, control, and shared documents (e.g., formalised design specifications), whereas ASD adopts informal communication and interpersonal collaboration. Therefore, GDAD efficacy is still arguable. This chapter has shown that successful GDAD projects have used some plan-driven principles, in general. Moreover, this chapter has shown that communication is the biggest challenge faces GDAD. In addition, this chapter has shed some light on the gaps of previous literature regarding GDAD communication. To sum-up the findings of this chapter, there is a big need for a systematic review that covers the current literature of GDAD successful projects, especially with regards to communication as the heart of agile development.

The next chapter, building on the current chapter, continues the background information review. It critically reviews the challenges of GDAD communication and the strategies and techniques used to mitigate the impact of these challenges in order to build a rigorous base that enables investigation of GDAD communication.

Chapter 3 Challenges of Communication in Geographically Distributed Agile Development

In Ch.2, most agile methods were reviewed and defined. XP was identified as the most used method within co-located local teams and Scrum was identified as the most used method in GDAD. Most organisations that used GDAD employ both XP and Scrum methods, principles and features at the same time. While Scrum features (e.g., Scrum Master, Product Owner, and Product Backlog) are used between distributed teams, XP features (e.g., Pair Programming, TDD) are used within the same co-located team. This combination between the two methods enhances the communication and control in GDAD environment. In addition, as discussed in Ch.2, communication is the most challenging issue in GDAD. Moreover, Ch. 2 identified a number of gaps that highlighted the need for systematically reviewing the literature to build the current state of the art about GDAD communication, which is the focus of this chapter.

3.1 Introduction

This chapter presents the second major stream of literature necessary for establishing the foundation of the study, drawn from academic and practitioner sources. This stream of literature focuses on GDAD communication challenges and the techniques to mitigate these challenges. Moreover, this stream fine-tunes the gaps found in the literature in Ch.2 regarding GDAD communication.

To identify the GDAD communication challenges and the techniques used to mitigate them, the current chapter covers the two phases of systematic literature review:

1. The first phase of the systematic literature review focuses on collecting and categorising the challenges that face GDAD, in general. The purpose of this phase is to see to what extent communication challenges impacts GDAD.

2. The second phase focuses on collecting, categorising, and analysing the challenges that face GDAD communication, as the focus of this study. The second phase is conducted in the following two stages. Firstly, a systematic review to communication challenges of DSD, in general (i.e. agile and traditional software development) is conducted. This review includes the empirical and non-empirical studies in the distributed environment. Secondly, a systematic review to GDAD communication is conducted. This review includes only the empirical studies that mention the communication issue in GDAD environment. Combining the results of both stages ensures better and deeper understanding to the issue of communication in GDAD environment.

This chapter progresses as follows. Section 3.2 discusses the challenges of GDAD. Sections 3.3 discuss the systematic review findings of GDAD communication challenges and the techniques to mitigate GDAD communication challenges. An overview of GDAD communication challenges is presented in Section 3.4. The motivation of the thesis is discussed in Section 3.5. The rationale for employing agile EA to enhance GDAD communication is discussed in Section 3.6. Finally, the chapter is summarised in Section 3.7.

3.2 GDAD Challenges

Globalisation and the high improvement in communication technologies have created a new type of business market competition and development (Herbsleb & Moitra 2001). This encourages software development organisations to adopt geographically DSD. DSD has some advantages over local development such as low cost, proximity to the market, quick formation of virtual teams, high quality, and to gather all talented developers around the world (Herbsleb 2007). The result of using DSD is multisite software development, multicultural teams, and a globally distributed undertaking. As a result, there are needs for high level arrangement, control and management, and communication technologies (Paasivaara & Lassenius 2003).

Organisations that use agile practices have started to combine and engage with deploying the DSD in order to gain the benefits of distributed development, as

mentioned above (Abrahamsson et al. 2009; Herbsleb 2007; Lanubile 2009). Despite the above lucrative benefits, GDAD also encompasses many challenges (Abrahamsson et al. 2009; Boehm & Turner 2003; Herbsleb & Moitra 2001; Holmstrom et al. 2006b; Niazi et al. 2013).

There is a growing body of literature that places emphasis on GDAD challenges. For example, a central view is that communication across distributed locations is a major source of problems in GDAD (Korkala & Maurer 2014; Shrivastava & Rathod 2015). GDAD faces problems of reduced coordination, collaboration, group awareness, knowledge management, project and process management, risk management, and process support (Jimenez et al. 2009). Other challenges are also identified including team size, team distribution, and project domain (Agerfalk et al. 2009). In addition, GDAD projects face problems due to strategic, cultural, geographic, and technical issues (Sharp & Robinson 2008). GDAD can also create a lack of close physical proximity, shared context and knowledge, team cohesion, and availability of team members (Paasivaara et al. 2009; Paasivaara et al. 2013). Ågerfalk et al. (2005) provide a high-level categorisation of GDAD problems through three major categories: communication, coordination, and control challenges. These categories arise from temporal (i.e. different time-zone), geographical (different physical locations), and socio-cultural distances (i.e. an actor's understanding of another actor's values and normative practices).

To reveal the whole punch of GDAD challenges and to state the position of GDAD communication issue among these challenges, we had conducted SLR for GDAD challenges using guidelines provided by Kitchenham and Charters (2007) and Rowe (2014). A customised literature searches and selection criteria was first developed and then applied to the selected electronic database. The following electronic databases to run this SLR were used: (1) IEEEXplore (www.ieeexplore.ieee.org/Xplore/), (2) ACM digital library (www.portal.acm.org/dl.cfm), (3) Elsevier Science Direct (www.sceincedirect.com/), (4) SpringerLink (www.springerlink.com/), and (5) Google Scholar (http://scholar.google.com.au/). The review included papers that focus on agile development and on traditional software development. The total number of selected studies for this review was 37 [S1–S37] papers (see Appendix 3.1).

Table 3.1: Challenges of GDAD

Challenge Category	Challenge	Author	No. and (%) of Studies
Communication Issues	• Formal/ informal • Team communication • Team coordination • Team collaboration • Team cooperation • Information/ knowledge sharing	Ali Babar et al. 2009, Avritzer et al. 2010, Berczuk 2007, Boden et al. 2007, Conchuir et al. 2009, da Silva et al. 2010, Estler et al. 2012, Helquist et al. 2011, Herbsleb 2007, Herbsleb & Mockus 2003, Herbsleb & Moitra 2001, Hossain et al. 2009b, Jiménez et al. 2009, Layman et al. 2006, Lee et al. 2011, Lee & Yong 2010, Mishra et al. 2012, Paasivaara et al. 2009, Paasivaara et al.2013, Ramesh et al. 2006, Sharp & Robinson 2008, Sharp et al. 2012	22 (59%)
Distance Differences	• Time-zones • Geographic	Agerfalk et al. 2009, Agerfalk et al. 2005, Balijepally et al. 2009, Batra et al. 2010, Boden et al. 2007, Conchuir et al. 2009, da Silva et al. 2010, Herbsleb & Mockus 2003, Herbsleb & Moitra 2001, Helquist et al. 2011, Holmstrom et al. 2006a, Jiménez et al. 2009, Layman et al. 2006, Lee et al. 2006, Lee & Yong 2010, Paasivaara & Lassenius 2010, Paasivaara et al. 2009, Paasivaara et al.2013, Ramesh et al. 2006, Sharp & Robinson 2008, Sharp et al. 2012	21 (57%)
People Differences	• Culture • Language • Trust • Attitudes • Values • Laws and regulations	Abufardeh & Magel 2010, Agerfalk et al. 2009, Agerfalk et al. 2005, Batra et al. 2010, Boden et al. 2007, Conchuir et al. 2009, da Silva et al. 2010, Estler et al. 2012, Helquist et al. 2011, Holmstrom et al. 2006a, Jiménez et al. 2009, Lanubile et al. 2003, Layman et al. 2006, Lee et al. 2006, Lee & Yong 2010, Paasivaara et al. 2009, Ramesh et al. 2006, Sharp & Robinson 2008, Sharp et al. 2012	19 (51%)
Technical Issues	• Tool support • Tracking and control • Process issues • Network • Software	Ali Babar et al. 2009, Avritzer et al. 2010, Boden et al. 2007, da Silva et al. 2010, Lanubile et al. 2003, Lee & Yong 2010, Hossain et al. 2009b, Jiménez et al. 2009, Sharp & Robinson 2008, Sharp et al. 2012	10 (27%)
Organisational Issues	• Organisational boundaries • Organisational functions	Ali Babar et al. 2009, Berczuk 2007, Boden et al. 2007, da Silva et al. 2010, Jiménez et al. 2009, Lanubile 2009, Lanubile et al. 2003, Lee et al. 2006, Sharp et al. 2012	9 (24%)
Management Issues	• Project domain • Multi-sites	Ali Babar et al. 2009, Batra et al. 2010, Boden et al. 2007, da Silva et al. 2010, Estler et al. 2012, Hossain et al. 2009b,	8 (21%)

	management • Schedule/ • Planning • Change management • Overall visibility • Overall quality • Confidentiality • Task allocation • Standards	Lee et al. 2011, Sharp & Robinson 2008	
Personal Skills	• Member skills differences • Unqualified member	Batra et al. 2010, Conboy et al. 2010, da Silva et al. 2010, Estler et al. 2012, Jiménez et al. 2009, Lee, Judge & McCrickard 2011	6 (16%)
Infrastructure Issues	• Infrastructure differences • Communication bandwidth • Office space	da Silva et al. 2010, Estler et al. 2012, Hossain et al. 2009b, Jiménez et al. 2009, Lee et al. 2011, Lee & Yong 2010	6 (16%)
Customer Communication	• Customer communication • Customer involvement	Korkala & Abrahamsson 2007, Korkala et al. 2010, Korkala et al. 2009, Layman et al. 2006, da Silva et al. 2010	5 (14)
Risk Issues	• Risk management	Batra et al. 2010, da Silva et al. 2010, Jiménez et al. 2009	3 (8%)

Data was extracted from the selected studies and categorised into 10 major categories using the Grounded Theory coding techniques (Glaser 1978). Tables (3.1 and 3.2) summarise these challenges along with the techniques/recommendations or strategies to overcome these challenges. Each category includes some challenges (e.g., Distance Differences category has "different time-zones" and "different geographic areas"). Here, we don't discuss these challenges and their definitions in detail as the detailed discussion will be provided in Sections 3.3, since the focus of this thesis is on studying GDAD communication. Both GDAD challenges and GDAD communication challenges were categorised under the same categories, in general.

Table 3.1 shows the GDAD challenges under each category. 10 categories and 39 challenges were identified. 19 studies have mentioned that the biggest issue that faces GDAD is the Communication, making up 59 % of the selected studies. The second mentioned category was the Distance Difference of 57 %. The third was People Difference category which was mentioned in 51 % of the selected studies. In fact,

Distance Difference and People Difference are the origin and the reason of all problems of GDAD (Agerfalk et al. 2009; Layman et al. 2006); however, the number and percentage of studies have been reported here as they were found in the systematic review. The minimum mentioned GDAD challenge was the Risk Issue, which makes only 8 % of the selected studies.

Table 3.2: Techniques/ solutions to GDAD challenges

Challenge Category	Recommendations
Communication Issues	Use architectural design, share high level goals, ensure team member's roles, enhance communication/ sharing mechanism, use documentation, use management support, promote team gathering, promote visit exchange, use tools and media networks, enhance synchronous working hours, use project tools, use code duplication rather than refactoring, use less control on communication flow, use well-defined work modules, promote informal interactions, use multiple communication modes, use communication Protocols, promote using common software process among sites, use detailed planning, outsourcing manager
Distance Differences	Ensure task's awareness, promote frequent visits between sites, use shared documents, use project management tools, use coping strategies (flexible and rigorous), enhance synchronicity (to set up meetings at times reasonable for most teams), use secure office space for local teams
People Differences	Ensure training on different cultures, use communication protocols, enhance work progress visibility, ensure that the outsourcing manager is part of distributed sites, ensure training in collaboration and coordination tools, use multiple communication modes, use configuration management, enhance using of wiki and photo gallery
Technical Issues	enhance using of media tools such as instant messaging and group chat, use different communication models, use best practices documentation, continuous integration (centralised), use integration tools, use configuration management, use management tools, use tracking tools, enhance visits exchange, enhance transition out of legacy process, use process refinements, use minimal up-front planning
Organisational Issues	Enhance engineering practice, enhance creating co-located teams, enhance distributed teams' communication, enhance pairing of development, enhance visit's exchange
Management Issues	Enhance mini-teams locally managed, enhance limited distributed sites, ensure synchronicity of meetings, enhance work synchronisation among sites, ensure secure IT infrastructure, enhance using of follow up system, use change management system, enhance visits exchanges, use "solar system" as a tool, maintain common quality standards among sites, ensure well-defined work modules, train teams on complementary skills and different cultures, ensure clear criteria for task allocation, use outsourcing manager role
Personal Skills	Enhance feedback, use documentation, enhance self-assignment, enhance training program, enhance cross-team observation/validation, enhance 'bought-in' developers, enhance team decision-making, promote performance evaluation, ensure hiring right people, use outsourcing manager role
Infrastructure Issues	Use multiple communication modes, enhance knowledge transfer mechanisms, enhance meeting in a single specific room always
Customer Communication	Ensure customer involvement, ensure not hiding information from customer
Risk Issues	Promote constant risk management, regular evaluation and assessment

Table 3.2 summarises the techniques/strategies or recommendations to mitigate the effect of the GDAD challenges. As shown in the table, the recommendations focus on using the available communication tools between distributed team members, synchronising working hours between distributed sites, and involving customer or customer representatives in the development. In addition, special focus is given to train members on other member's cultures. However, other studies prefer using the plan-driven principles in GDAD environment such as using documentation, applying control and tracking tools among distributed sites.

As shown in Table 3.1, the communication issue was the most reported problem of GDAD. Hence, this thesis focuses on studying the GDAD communication issue in order to provide an approach or model to enhance GDAD communication. To do so, this thesis also focuses on the techniques, strategies, recommendations, and solutions that have been developed by the previous authors or practitioners to mitigate GDAD communication challenges. Therefore, the following section discusses, in detail, all challenges and causes of GDAD communication problem as well as the techniques, strategies, recommendations, and solutions through the systematic review of empirical studies that discussed GDAD. It also shed more light on the impact of communication on the software performance (success), and on communication patterns in GDAD environment.

3.3 GDAD Communication Challenges-SLR[1]

As it was mentioned in the introduction, two SLR were conducted to GDAD communication challenges. The first review included studies from GDAD and traditional DSD either empirical or expertise reports. The second review included only GDAD empirical studies. The findings of both reviews were very similar. GDAD communication challenges in both reviews, generally, were categorised under the same categories. However, 6 categories and 21 GDAD communication challenges were identified in the first review, while in the second review, 6 categories and 17 GDAD communication challenges were identified (see Table 3.11). Here, the second review will be discussed since it focuses on GDAD empirical studies only, which give more

[1] This section appeared in Alzoubi et al. 2016

insight intoto the GDAD communication issue. Moreover, some challenges which were found in the first review relate to traditional DSD and not to GDAD and hence, it is more pragmatic to focus on the more related studies (i.e. empirical GDAD). Finally, the previous authors emphasised the need of empirical evidence of how agile practices enhance GDAD communication and how GDAD communication challenges can be mitigated (e.g., Agerfalk et al. 2009; Hummel et al. 2013). Therefore, this review attempts to shed more light on the empirical studies conducted in the field of GDAD communication and thereby identify the practical GDAD communication challenges and relevant mitigation techniques. For more details about the first review, please refer to (Alzoubi & Gill 2014).

3.3.1 SLR Method and Limitation

The main aim of this review was to identify the challenges of GDAD communication, and to find out how empirically agile practices enhance GDAD communication and how GDAD communication challenges can be mitigated. Non-empirical (e.g., theoretical and conceptual) studies are beyond the scope of this review. This may limit the findings of the review since some practitioners' reports and papers that discuss individual experience were excluded from the review process, which may have some important information (Rowe 2014). Hence, this review focused on answering the first two research questions:

RQ1: What are the challenges or factors that limit GDAD communication?

RQ2: Which techniques have been used to overcome these challenges and enhance GDAD communication?

This review adopted SLR approach (Kitchenham & Charters 2007; Rowe 2014) to identify and synthesise GDAD communication challenges. The SLR is a structured and systematic approach to identifying, selecting and synthesizing recent literature relevant to the research question (Kitchenham & Charters 2007). Additionally, to ensure the quality of this review, the guidelines for citation and evaluation procedures (Dyba & Dingsoyr 2008; Rowe 2014) that complement the original SRL (Kitchenham & Charters 2007) approach were also used. SLR has been applied in distinct stages: (1) selection of

inclusion and exclusion criteria, (2) selection of data sources and search strategies, (3) citation and inclusion decision management, (4) final selection and quality assessment, and (5) data extraction and synthesis. These stages are discussed in the following Sections (3.3.1.1 – 3.3.1.5). Applying customised search criteria derived from the research questions, 21 relevant empirical studies were identified and reviewed (see Appendix 3.2). The findings of this review are discussed in Section 3.3.2.

3.3.1.1 Inclusion and Exclusion Criteria

In this review, we included only those studies that presented empirical data on GDAD and passed the minimum quality criteria discussed in Section 3.3.1.4 (Table 3.5). Therefore, studies were excluded if their focus was not on GDAD or if they did not present empirical data. This review included studies up to July 2014; qualitative, quantitative or mixed measurement studies; small, medium and large GDAD organisations; and both professional and academic experimental software projects. Only papers written in English were included. Furthermore, as our research questions are concerned with GDAD communication (irrespective of any specific agile method), studies that focused on single methods, mixed-methods (i.e., Scrum and XP) or agile methods in general were included.

3.3.1.2 Data Sources and Search Strategies

The following five well-known electronic databases were used in this review.

1. IEEE Xplore (www.ieeexplore.ieee.org/Xplore/).
2. ACM Digital Library (www.portal.acm.org/dl.cfm).
3. Elsevier ScienceDirect (www.sceincedirect.com/).
4. SpringerLink (www.springerlink.com/).
5. Google Scholar (http://scholar.google.com.au/).

These databases are assumed to provide sufficient literature coverage for this review. In addition, the following relevant and important agile conference proceedings discussing the use of GDAD were manually searched and included in this study: XP, XP/ Agile Universe, and Agile Development Conference.

The reviewed papers come from industrial, qualitative/quantitative and experimental academic studies. Figure 3.1 shows the review process and the number of papers identified at each stage. Table 3.3 shows the terms and keywords used to run the first stage of the search. XP and Scrum terms were included in the search terms because these two are the most used agile methods (Dingsoyr et al. 2012). In this stage, each item from the first category (i.e., ‘‘Communication Practice’’) was combined with each item from the second category (i.e., ‘‘Distributed Software Development’’) and each item from the third category (i.e., ‘‘Use of Agile Practice’’) using the Boolean ‘‘AND’’ operator, which finds all articles on ‘‘Communication Practices’’, ‘‘Distributed Software Development’’ and ‘‘Use of Agile Practice’’. That is, we searched every possible combination of one item each from the first category AND the second category AND the third category. The search excluded articles that addressed discussion comments, prefaces, editorials, news, article summaries, reviews, correspondences, discussions, reader’s letters, and summaries of tutorials, workshops, panels, and poster sessions. This search strategy resulted in a total of 1587 ‘‘hits’’; however, after excluding the duplicate papers, the number of hits dropped to 799.

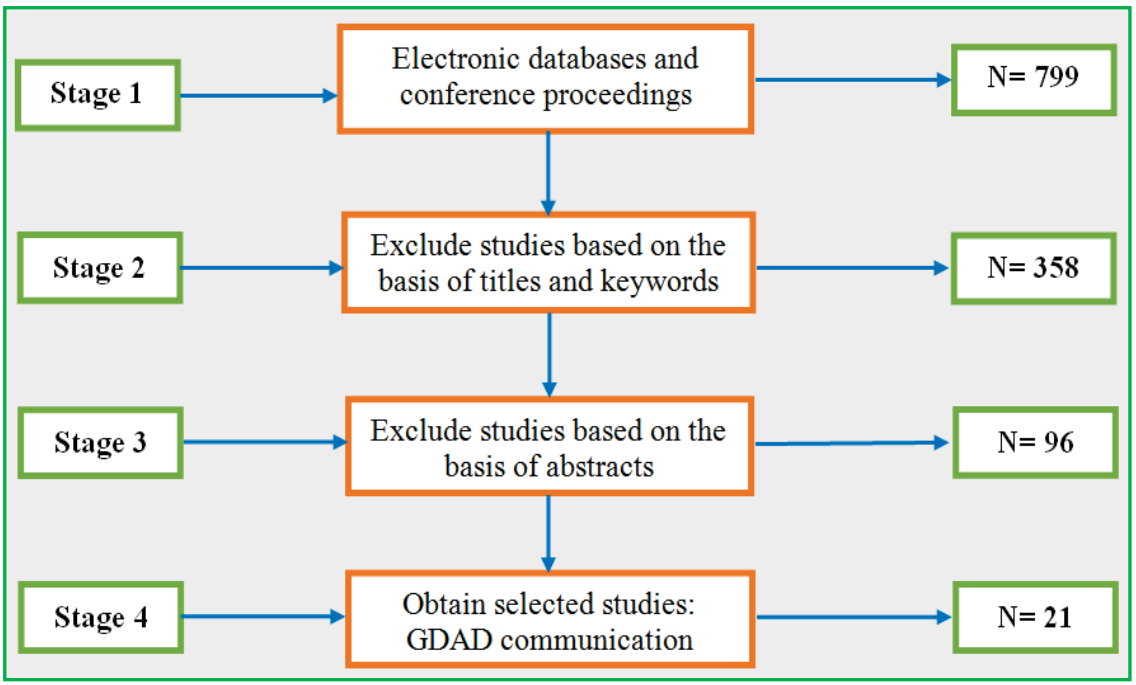


Figure 3.1. GDAD communication SLR selection process

Table 3.3: GDAD communication SLR search terms

Search Category	Keywords
Communication Practice	Communication, Team communication, Cross-team communication, Offshore communication, Outsourcing communication, Customer communication, Social media communication, Communication tool, Communication technology
Distributed Software Development	Distributed agile, Multi-sites agile, Global agile development, Multi-team agile, Global software development, Distributed development, Distributed development teams, Global software development, Global development, Global software engineering, Offshore development, Off shoring, Outsourcing development, open source, near shore, near shoring, Multi-sites development, Global software engineering, Dispersed teams
Use of Agile Practice	Agile, Agile methods, Agile practice, Scrum, Scrum practice, Scrum method, XP, Extreme Programming, XP practice, XP method

3.3.1.3 Managing Citation and Inclusion Decisions

This review followed the citation procedure reported by Dyba˚ and Dingsøyr (2008). EndNote was used to store the relevant citations from stage 1 (N = 799). The citations were then imported into Excel sheets, where the source of each citation and subsequent inclusion/exclusion decision were recorded. EndNote databases and *Excel* sheets were separately established for each stage. In this stage, the titles of all 799 studies were reviewed and excluded studies that were clearly not about GDAD communication. In some cases, the title failed to clearly identify whether the study was within the scope of this review. In this case, the articles were included in the next review stage. In this stage, we identified 358 relevant studies. Table 3.4 summarises the assessment method and criteria for each stage.

At stage 3, the studies that did not focus on GDAD communication and did not present empirical data were excluded by scanning the abstracts. Some abstracts were misleading, gave little indication of the content of the full paper or did not clearly indicate whether the study was indeed empirical. Therefore, at this stage, all studies that indicated some form of GDAD experience were included. In this stage, 262 articles were excluded, leaving 96 articles for the final selection stage.

Table 3.4: GDAD communication SLR assessment method

Filtration Stage	Method	Assessment Criteria
1st Filtration	Identify relevant studies form searched databases	Search keywords (all items)
2nd Filtration	Exclude studies on the basis of titles	Title = search term (s) Yes = accepted No = rejected
3rd Filtration	Exclude studies on the basis of abstracts	Abstract = communication Yes = accepted No = rejected
Final Filtration	Obtain selected papers and critically appraise studies	Address GDAD communication Discuss empirical research (Yes= accepted; No = rejected)

3.3.1.4 Final Selection and Quality Assessment

If the study did not clearly indicate GDAD communication in the title, abstract, and keywords, it was included in a detailed final quality assessment. At the final stage, screening criteria were used to ensure the relevance and quality of the selected study.

Table 3.5: GDAD communication SLR quality criteria of selected studies (Dyba˚ & Dingsøyr 2008)

Quality Criteria
1. Is the paper based on research?
2. Is there a clear statement of the aims of the research?
3. Is there an adequate description of the context in which the research was carried out?
4. Was the research design appropriate to address the aims of the research?
5. Was the recruitment strategy appropriate to the aims of the research?
6. Was there a control group with which to compare treatments?
7. Was the data collected in a way that addressed the research issue?
8. Was the data analysis sufficiently rigorous?
9. Has the relationship between researcher and participants been considered to an adequate degree?
10. Is there a clear statement of findings?
11. Is the study of value for research or practice?

The following screening criteria were used to ensure that the studies address the research topic, as shown in Table 3.5. The criteria were adopted from Dyba˚ and Dingsøyr (2008). These 11 criteria ensure the quality, relevance, credibility and rigour of the studies used in this research (Dyba˚ & Dingsøyr 2008):

- Three screening criteria were used to ensure the quality of the reviewed study by identifying the study's rationale (i.e., reported empirical research), aims (i.e., clearly reported aims and objectives), and context (i.e., the context in which the study was carried out), as shown in the first three questions of Table 3.5. These criteria were used to ensure that only empirical studies were included and represent the minimum quality threshold of the selected studies. Only criterion 1 was used as a basis for including or excluding a study in the final stage.

- Five criteria were used to ensure that the study was rigorous (i.e., used thorough and appropriate approaches for collecting and analysing data). The following criteria were used to assess the rigour of the selected studies (questions 4–8 in Table 3.5):
  1. The study used an appropriate research design to address its aims.
  2. The study adequately described the sample used and the methods for identifying the sample.
  3. The study used control groups to compare treatments.
  4. The study used appropriate data collection methods.
  5. The study adequately described the data analysis methods.

- Two criteria were used to ensure the credibility (i.e., the validity and meaningfulness of the findings) of the selected studies (questions 9 and 10 in Table 3.5).
  1. Relationship between the researcher and participants was considered to an adequate degree.
  2. The study provided clear findings with credible results and justified conclusions.

- Finally, one criterion was used to ensure the relevance (usefulness for GDAD industry and research community) of the selected studies (question 11 in Table 3.5) by identifying whether the study provided value for practice and/or research.

After applying the first criterion, only 21 studies out of the 96 studies passed stage 3 and were included in the analysis stage. Some papers that were published in different years but used the same empirical data were excluded. For instance, the studies (Paasivaara & Lassenius 2010) and (Paasivaara et al. 2009) used the same empirical data, as did studies (Hossain et al. 2011) and (Bannerman et al. 2012). In this case, only the study most related to this review (i.e., Bannerman et al. 2012; Paasivaara et al. 2009) was included in the extraction and synthesis stage. Table 3.6 summarises the number of selected studies in each stage. Most studies were found in the ‘‘IEEE Xplore’’ and ‘‘SpringerLink’’ databases, comprising 33% of the selected studies for each. Three studies were found in the ‘‘Google Scholar’’ database, representing14% of the selected studies. Only 2 studies each were found in the ‘‘ACM Digital Library’’ and ‘‘ScienceDirect’’ databases.

Table 3.6: GDAD communication SLR search results

Database	1st Filtration	2nd Filtration	3rd Filtration	Selected Studies	Percent
IEEEXplore	328	119	39	7	33%
ACM Digital library	167	97	22	2	10%
SpringerLink	125	83	20	7	33%
Google Scholar	104	37	8	3	14%
Science Direct	75	22	7	2	10%
Total	799	358	96	21	100%

In each stage of the review, two researchers searched independently, and a third researcher always accompanied each researcher to review the findings separately from the other researcher. At the end of each stage, all authors sat together and discussed the agreement on the number of studies to be included in the next stage. All disagreements were solved by the three researchers’ meeting (all hands in meetings). For example, in the final stage (i.e., obtain selected study) we found 82 (85%) agreements among 96 assessments. All disagreements were resolved by the three researchers through rigorous discussions. As a result of the discussions, another set of 61 articles was excluded at this stage, leaving a final set of 21 (see Appendix 3.2) articles for data extraction and synthesis.

3.3.1.5 Data Extraction and Synthesis

Communication Common Ground theory (Clark & Brennan 1991) and the Unified Model of Information Software Development Success (Siau et al. 2010) were used to develop the theoretical framework (Figure 3.2). Communication common ground refers to the mutual, common or joint knowledge and the beliefs and suppositions held by the communication parties. Communication common ground enables successful coordination, as it allows interdependent actors to adjust their actions in a manner appropriate for the other actors. Common ground facilitates the delivery of an understandable message with minimum effort (i.e., efficient and effective communication). Communication common ground can be enhanced by using different communication techniques based on the communication medium. It also identifies constraints (i.e., challenges) that decrease communication efficiency and communication effectiveness. Thus, common ground offers a comprehensive view of the communication process.

A deductive approach was followed based on the themes provided by The Unified Model of Information Software Development Success (Siau et al. 2010). Based on a comprehensive literature review, this model synthesises past research efforts in the software development field and identifies important factors that influence the software development process (e.g., team factors, organisational factors and human factors). This model is suitable for coding our findings as it provides sufficient coverage of the software development process as well as its input and output factors. The combination of common ground and the Unified Model provided us with a comprehensive framework to evaluate GDAD communication.

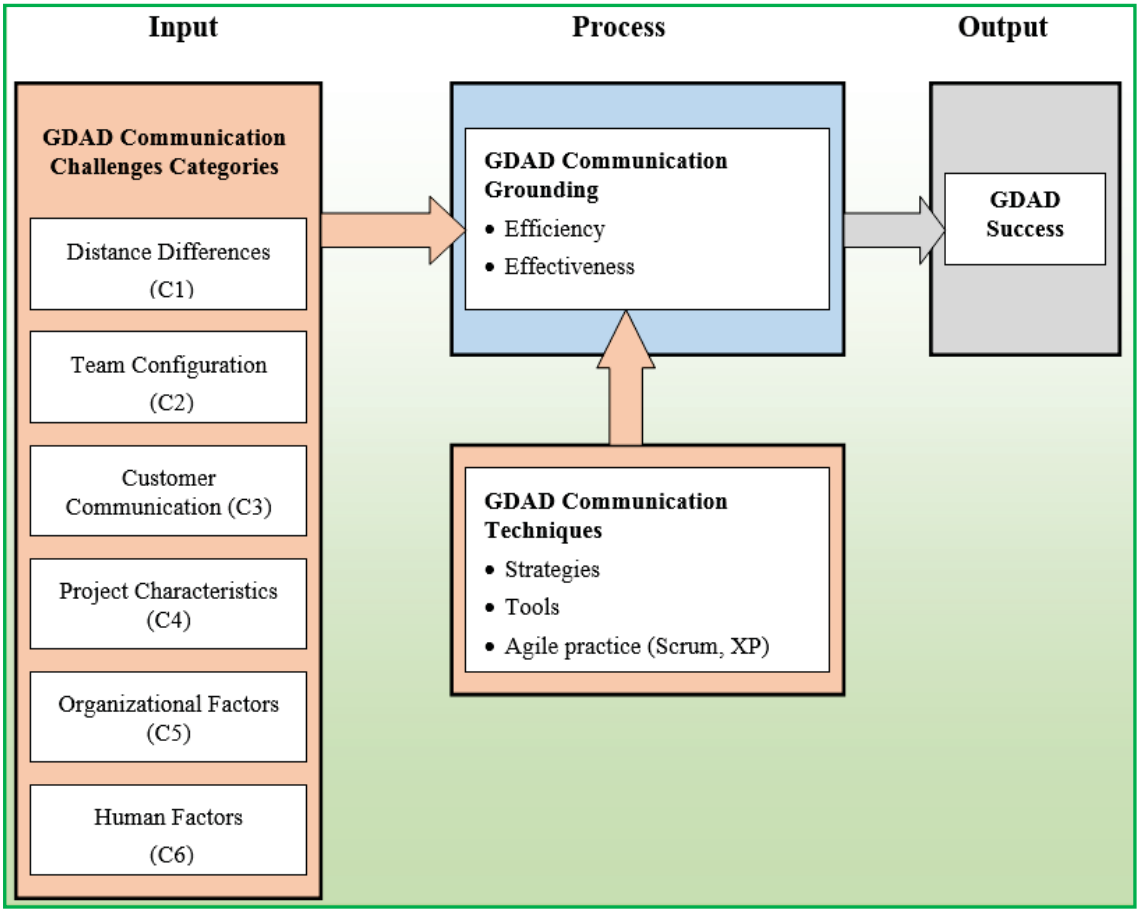


Figure 3.2. Theoretical framework and selected categories of GDAD communication SLR

3.3.2 SLR Findings

21 [S1–S21] empirical studies on GDAD were identified and reviewed using the overarching SLR approach described in the previous section; see (Appendix 3.2). First, an overview of the selected studies is provided. Next, the characteristics of the selected studies, the research methods, and the quality of these studies are discussed. Finally, the analysis of the selected studies to answer the identified research questions (i.e., to identify GDAD communication challenges and techniques/strategies and recommendations to overcome these challenges) are discussed.

3.3.2.1 Overview of Selected Studies

This SLR found that all of the selected studies were conducted in or after 2006. With respect to the types of agile methods studied, it can be observed from Table 3.7 that 8 (37%) of the studies in this review did not mention any specific agile method (i.e., the

agile method used was not identified). Meanwhile, 33% of the selected studies were relevant to the Scrum method. The mixed Scrum and XP method was studied in four empirical studies (20%). Finally, only two studies (10%) applied the XP method, representing the smallest category.

Table 3.7: Agile method used in the GDAD communication SLR selected studies

Agile Method	Number	Year	Percent	Author
Scrum	7	2008 - 2013	33%	Bannerman et al. 2012; Paasivaara et al. 2013; Paasivaara et al. 2009; Sindhgatta et al. 2011; Valimaki & Kaariainen 2008; Vallon et al. 2013; Zieris & Salinger 2013
XP	2	2006 - 2010	10%	Layman et al. 2006; Maruping 2010
Mix (Scrum, XP)	4	2006 - 2012	20%	Holmstrom et al. 2006b; Korkala & Maurer 2014; Lee & Yong 2010; Sharp et al. 2012
Not-identified (General)	8	2007 - 2014	37%	Boden et al. 2007; Bosch & Bosch-Sijtsema 2010; Dorairaj et al. 2011; Gill & Bunker 2013; Green et al. 2010; Korkala & Abrahamsson 2007; Korkala et al. 2010; Yadav et al. 2007

Analysis of the selected studies revealed a strong focus on using the Scrum method in the GDAD environment. The reason may be due to the project management and coordination roles, artefacts and practices offered by Scrum (Sutherland et al. 2007). For instance, Scrum includes a Product Owner role, which can facilitate communication requirements among GDAD teams. Scrum artefacts include the Product Backlog and Scrum Backlog, which can also be scaled to support GDAD (Sutherland et al. 2009). On the other hand, the XP method was more preferred and used for development by co-located teams (e.g., pair programming). In many cases, the mixed Scrum and XP method was used by GDAD teams to support project communication and management as well as project development.

Table 3.8 provides an overview of the selected studies according to the publication channel. The single largest publication channel was found to be the ‘‘ICGSE’’ and ‘‘Agile Processes in Software Engineering and Extreme Programming’’, which published three studies each. Ten studies were published in conferences. Six studies were published as book chapters, and only five empirical studies (relevant to the research questions in hand) appeared in scientific journals.

Table 3.8: Publication channel of GDAD communication SLR selected studies

Publication Channel	Type	Study	Number
Information and Software Technology	Journal	Layman et al. 2006	1
Information Systems Management	Journal	Holmstrom et al. 2006b	1
JSS	Journal	Korkala & Maurer 2014	1
VINE	Journal	Gill & Bunker 2013	1
Empirical Software Engineering	Journal	Lee & Yong 2010	1
EUROMICRO SEAA	Conference	Korkala & Abrahamsson 2007	1
HICSS	Conference	Bannerman et al. 2012	1
ICGSE	Conference	Boden et al. 2007; Paasivaara et al. 2009; Zieris & Salinger 2013	3
ICSE	Conference	Paasivaara et al. 2013	1
PACIS	Conference	Yadav et al. 2007	1
International Conference on Product Focused Software	Conference	Korkala et al. 2010	1
SPLASH	Conference	Sindhgatta et al. 2011	1
WASET	Conference	Green et al. 2010	1
Agility Across Time and Space	Book section	Bosch & Bosch-Sijtsema 2010; Maruping 2010	2
Agile Processes in Software Engineering and Extreme Programming	Book section	Dorairaj et al. 2011; Sharp et al. 2012; Vallon et al. 2013	3
Enterprise interoperability III	Book section	Valimaki & Kaariainen 2008	1

Regarding the year of publication, we found no empirical studies on GDAD prior to 2006 that matched our selection criteria. We found that two empirical research studies were published in 2006, three in 2007, two in 2009, four in 2010, two in 2011, two in 2012, four in 2013, and only one study each in 2008 and 2014. These numbers indicate that a focus on GDAD study was more prevalent during 2010 and 2013. This review searched databases up to July 2014, which may explain the small number of papers published in 2014 (i.e., only one study).

3.3.2.2 Methodological Quality of Selected Studies

As mentioned in Section 3.3.1.4, we assessed the quality of each of the selected studies according to 11 quality criteria (Table 3.5) adopted from Dyba˚ and Dingsøyr (2008). These 11 criteria provided a measure of our confidence that a selected study could make

a valuable contribution to our review. Each of the 11 criteria was graded on a dichotomous (‘‘yes’’ or ‘‘no’’) scale. As mentioned above, the inclusion of the studies was based on the first assessment criterion (i.e., the study should report empirical research). The results of the quality assessment are shown in Table 3.9.

Table 3.9: Quality assessment of GDAD communication SLR selected studies

Study	1	2	3	4	5	6	7	8	9	10	11	Total
	Research	Aim	Context	Research design	Sampling	Control group	Data collection	Data analysis	Reflexivity	Findings	Value	
S1	1	1	1	1	0	0	1	1	0	1	1	8
S2	1	1	1	1	1	0	1	1	0	1	1	9
S3	1	1	1	0	0	0	1	0	0	1	1	6
S4	1	1	1	1	1	0	1	1	0	1	1	9
S5	1	1	1	1	1	0	1	1	1	1	1	10
S6	1	1	1	1	0	0	1	0	0	1	1	7
S7	1	1	1	1	0	0	1	1	1	1	1	9
S8	1	0	1	1	0	1	1	0	0	1	1	7
S9	1	1	1	1	1	1	1	1	1	1	1	11
S10	1	1	1	1	0	0	1	0	0	1	1	7
S11	1	1	1	1	0	0	1	0	0	1	1	7
S12	1	1	1	0	0	0	1	1	0	1	1	7
S13	1	1	1	0	0	0	1	0	0	1	1	6
S14	1	1	1	1	0	0	1	1	0	1	1	8
S15	1	1	1	1	0	0	1	1	0	1	1	8
S16	1	1	1	0	0	0	1	0	0	1	1	6
S17	1	1	1	0	0	0	0	1	0	1	1	6
S18	1	1	1	1	1	0	1	1	0	1	1	9
S19	1	1	1	1	1	0	1	1	0	1	1	9
S20	1	1	1	0	0	0	1	1	0	1	1	7
S21	1	1	1	0	0	1	1	1	0	1	1	8
Total	21	20	21	14	6	3	20	14	3	21	21	

Because we only included research papers in our review, all selected studies were graded as ‘‘1’’ on the first screening criterion (first column). With regards to the study’s aim, only one study was found not to have a clear statement of its aims. The third criterion (context of the study) showed that all studies included some form of description of the context in which they were carried out. For the chosen research design, seven studies did not adequately report the research design used to achieve the aims of the research. As many as fifteen out of the twenty-one studies did not clearly report the recruitment strategy (i.e., sampling) used to achieve their stated aims. Only three studies reported using other groups or baselines with which to compare their findings. Only one study did not describe its data collection method. Seven studies did not report their data analysis procedures. Only three studies reported how they avoided

biasing the findings. It was found that research bias issues, recruitment strategy and control group assessment were often not reported in the selected studies. This may be related to the fact that most of the selected studies were conference papers (see Table 3.8), as conference papers generally report fewer details about the research methods. The overall quality of the selected studies score was greater than 7.8, which is satisfactory and above 6 (the minimum passing score) (Dyba˚ & Dingsøyr 2008). One study was graded 11 (full score), and none obtained negative scores in the quality assessment. The highest number of negative scorings was five. Thus, overall, all selected studies were of acceptable quality for this study and added value to the body of knowledge. However, these findings also evidence the need for more quality empirical studies in the area of GDAD communication.

3.3.2.3 Scope of Selected Studies

The selected studies varied in their scope, as shown in Table 3.10, which can be categorised as follows: (i) the challenges of adopting GDAD (several studies), (ii) the scale-up of local agile development to a GDAD environment, (iii) the successful implementation of agile methods in GDAD, (iv) the challenges of GDAD communication, and (v) solutions to GDAD challenges in general and GDAD communication challenges in particular. As the focus in this study is on GDAD communication challenges and solutions, the findings were derived from those scopes that included GDAD communication. To this end, we categorised our findings into two broad categories: GDAD communication challenges and techniques to overcome GDAD communication challenges. By investigating these two research questions, we aim to provide a synthesis of the literature of GDAD communication challenges and techniques for both researchers and practitioners.

Table 3.10: GDAD communication SLR selected studies aims

Study	Ref.	Study aim
Bannerman et al. 2012	S1	Report the impact of temporal, geographical and sociocultural distance and seven Scrum practices on the use of Scrum practices in global software development (GSD)
Boden et al. 2007	S2	Show to what extent small and medium enterprises rely upon situated coordination practices in order to warrant their agility
Bosch & Bosch-Sijtsema 2010	S3	Study the relation between large-scale and agile approaches to software development as well as current problems
Dorairaj et al. 2011	S4	Explore distributed agile software development communication challenges from the perspective of agile practitioners.
Gill & Bunker 2013	S5	Investigate and present the developers' needs for communication technologies in the context of distributed adaptive development environments (DADE)
Green et al. 2010	S6	Explore the relevancy of communication richness in various agile in distributed agile development phases and its impact on quality
Holmstrom et al. 2006b	S7	Explore how agile practices can reduce three kinds of "distance": temporal, geographical, and sociocultural in global software development (GSD)
Korkala & Abrahamsson 2007	S8	Compare case study findings against existing recommendations about communication in distributed agile development
Korkala & Maurer 2014	S9	Identify non-value producing communication elements and solutions to mitigate them
Korkala et al. 2010	S10	Describe results from a case study of customer communication in a globally distributed product development program applying both traditional and agile approaches
Layman et al. 2006	S11	Understand how globally-distributed team created a successful project in a new problem domain using a methodology that is dependent on informal, face-to-face communication
Lee & Yong 2010	S12	Highlight successful practices and challenges that have been overcome by the globalization project, and suggest a framework for software globalization project management using a distributed Agile approach
Maruping 2010	S13	Outline some of the strategies and challenges associated with implementing agile methods in distributed software project teams
Paasivaara et al. 2013	S14	Describe distributed Scrum augmented with best practices in global software engineering (GSE)
Paasivaara et al. 2009	S15	Present findings of successfully adopted project in distributed Scrum
Sharp et al. 2012	S16	Report on a field study of one successful partially dispersed agile team
Sindhgatta et al. 2011	S17	Study communication characteristics of team members in a large, globally DSD project that uses the IBM Jazz platform (Scrum)
Valimaki & Kaariainen 2008	S18	Find the best practices to distributed Scrum
Vallon et al. 2013	S19	Investigate an agile approach from a real-world project in a distributed development environment
Yadav et al. 2007	S20	Report an exploratory quasi-experimental study, which investigates the performance of requirements analysis projects in an 'agile-rigid' distributed environment
Zieris & Salinger 2013	S21	Present the actual Scrum process and practices of the external teams and contrast them to the intended way of proceeding

3.3.2.4 GDAD Communication Challenges

As mentioned above, the Unified Model of Information Software Development Success (Siau et al. 2010) was used to derive and code the categories used for structuring and analysing the results of the selected studies.

As discussed in Ch.2, GDAD refer to ‘‘locally distributed’’ (i.e. in different physical locations within the same country) or ‘‘globally distributed’’ (i.e., affected by global factors, such as culture and language) agile development (Mishra et al. 2012; Ramesh et al. 2006). Some studies and recommendations identify three main communication challenges of agile development in general: Team Distribution, Team Size, and Project Characteristics (e.g., Hummel et al. 2013). However, because we are investigating the GDAD environment, the Team Distribution category will be considered through four separate categories (i.e., Distance Differences, Customer Communication, Organisational Factors and Human Factors). This helps to distinguish between the two types of GDAD (i.e., ‘‘locally distributed’’ and ‘‘globally distributed’’) and identify the challenges of these categories in terms of the big issues for GDAD communication, such as national culture and organisational culture (Agerfalk et al. 2009; Dingsoyr et al. 2012; Keskin 2009). We also refer to the Team Size category as Team Configuration (Agerfalk et al. 2009) as a more comprehensive definition. While both ‘‘locally distributed’’ and ‘‘globally distributed’’ agile development shares the first 4 categories of the communication challenge categories (i.e., Distance Differences, Team Configuration, Customer Communication, and Project Characteristics), categories 5 and 6 (i.e., Organisational Factors and Human Factors) may only be experienced in ‘‘globally distributed’’ agile development communication (Figure 3.2 and Table 3.11).

The input of the GDAD communication theoretical framework is represented by the GDAD communication challenge categories (C1–C6). These categories will negatively affect the GDAD communication grounding process, which negatively reflects on communication efficiency and communication effectiveness (Clark & Brennan 1991). The GDAD communication grounding process is enhanced by applying certain techniques that improve communication efficiency and effectiveness (Clark & Brennan 1991). These techniques include strategies, tools, and agile practices. The Scrum and

XP methods were given special focus in the analysis because, as mentioned previously in this chapter, they are considered as the most widely used agile methods in industry (Dingsoyr et al. 2012). The output of the GDAD communication grounding process is the GDAD success, which can be represented by such factors as project success (high performance) and user satisfaction (Agrawal & Chari 2007; Aladwani 2002; Siau et al. 2010).

Table 3.11: GDAD communication challenges and categories

Ref.	Category	GDAD Communication Challenges
C1	Distance Differences	1. Time-zone differences 2. Geographic differences
C2	Team Configuration	3. Team size 4. Number of teams 5. Coordination
C3	Customer Communication	6. Customer involvement 7. Customer representative involvement
C4	Project Characteristics	8. Project domain 9. Project architecture
C5	Organisational Factors	10. Project management process 11. Communication tools 12. Communication infrastructure 13. Organisational culture
C6	Human Factors	14. Language 15. National culture 16. Trust in teams or team members 17. Personal practice

To answer the first research question, the data were analysed and interpreted from the selected empirical studies and identified the categories of GDAD communication challenges (Figure 3.2 and Table 3.11). Each category has one or more challenges. A total of 17 GDAD communication challenges were identified.

Table 3.12 shows the challenge categories and the statistics related to each category. The identified challenges may become issues (i.e., negative impacts) and limit GDAD communication. These impacts are also discussed in the following paragraphs and are summarised in Table 3.13. These impacts were summarised under each concept category, as for GDAD communication challenges (Table 3.11), and extracted from the same studies for each category.

Table 3.12: GDAD communication challenges categories statistics

Ref.	Category	Frequency	Percentage	Selected Studies
C1	Distance Differences	16	76%	Bannerman et al. 2012; Boden et al. 2007; Bosch & Bosch-Sijtsema 2010; Dorairaj, Noble & Malik 2011; Green et al. 2010; Holmstrom et al. 2006b; Korkala & Abrahamsson 2007; Korkala et al. 2010; Layman et al. 2006; Lee & Yong 2010; Maruping 2010; Paasivaara et al. 2013; Paasivaara et al. 2009; Sharp et al. 2012; Sindhgatta et al. 2011; Zieris & Salinger 2013
C2	Team Configuration	10	48%	Boden et al. 2007; Bosch & Bosch-Sijtsema 2010; Dorairaj, Noble & Malik 2011; Green et al. 2010; Maruping 2010; Paasivaara et al. 2013; Sindhgatta et al. 2011; Valimaki & Kaariainen 2008; Vallon et al. 2013; Yadav et al. 2007
C3	Customer Communication	4	19%	Korkala & Maurer 2014; Korkala, Pikkarainen & Conboy 2010; Layman et al. 2006; Paasivaara et al. 2013
C4	Project Characteristics	2	10%	Bannerman et al. 2012; Bosch & Bosch-Sijtsema 2010
C5	Organisational Factors	11	52%	Bosch & Bosch-Sijtsema 2010; Dorairaj et al. 2011; Gill & Bunker 2013; Green et al. 2010; Holmstrom et al. 2006b; Korkala & Abrahamsson 2007; Korkala & Maurer 2014; Layman et al. 2006; Paasivaara et al. 2013; Vallon et al. 2013; Zieris & Salinger 2013
C6	Human Factors	10	48%	Bannerman et al. 2012; Boden et al. 2007; Bosch & Bosch-Sijtsema 2010; Dorairaj et al. 2011; Holmstrom et al. 2006b; Layman et al. 2006; Lee & Yong 2010; Paasivaara et al. 2013; Paasivaara et al. 2009; Zieris & Salinger 2013

3.3.2.4.1 C1-Distance Differences

This category refers to differences in the time-zones and geographical contexts of the GDAD teams. Two common characteristics of distance differences have been defined in the context of GDAD: time (temporal) and geographical distances (Agerfalk et al. 2005).

Temporal distance is ‘‘a directional measure of the dislocation in time experienced by two actors wishing to interact. Temporal distance can be caused by time zone difference or time shifting work patterns’’ (Agerfalk et al. 2005, p. 3).

Table 3.13: Impact of GDAD communication challenges on communication process

Ref.	Category	Impacts
C1	Distance Differences	Minimum communication opportunity, lack of face-to-face and informal communication, communication delay, long-time communication using technology, longer time of meetings, group meeting is difficult outside working hours (overlaps), lose track of the overall work process, synchronisation problem due to different religious holidays, the availability of developer difficulties, integration, coordination and communication cost for face-to-face meetings increase, lack of task awareness, lack of trust
C2	Team Configuration	Early communication difficulties, team member's configuration formation difficulties, getting all members to communicate with others, unwillingness to communicate to other team members, less understanding to teamwork, slowing down of the communication speed and pair programming
C3	Customer Communication	Less frequent communication with customer, weak relationship with customer, customer's un-involvement, hiding information from customer, miscommunication of requirements that leads to either developer used their experience or guess the requirements
C4	Project Characteristics	Misunderstanding or an unnecessary flow of communication due to the definition of a system and software structure, unnecessary communication, not match other sites or team's processes, lack of communication opportunities
C5	Organisational Factors	Team suffer, less business value and product quality, more constraint and risks, more organisational modes, less access control, more semantic interoperability, more contingency and disaster recovery plans, more communication channels, extra cost on training teams to different tools, technical and organisational culture incompatibilities between sites, lack of trust
C6	Human Factors	Misunderstanding and miscommunication difficulties, different holidays among distributed team members, inconsistency in work place, longer time for sharing, communication difficulties due to language, limited communication among GDAD teams, interpretation problem, silence of some participants

Geographical distance is ‘‘a directional measure of the effort required for one actor to visit another at the latter's home site. Geographical distance is best measured in ease of relocating rather than in kilometres’’ (Agerfalk et al. 2005, p. 3).

This category has been heavily referenced in the literature and is represented in the highest proportion (i.e., 76%) of the selected studies. Distance Differences have been reported as the main challenge of GDAD communication (Korkala & Maurer 2014; Korkala et al. 2010). The selected studies reported that Distance Differences adversely impact communication Common Ground between GDAD teams or team members (e.g.,

Dorairaj et al. 2011; Holmstrom et al. 2006b; Paasivaara et al. 2009). Distance Differences minimise communication efficiency and effectiveness between GDAD teams and decrease the opportunity for groups to organise meetings outside standard local working hours (Dorairaj et al. 2011; Paasivaara et al. 2009). In addition, Distance Differences increase the cost and logistics of holding face- to-face meetings for GDAD teams (Bosch & Bosch-Sijtsema 2010). Distance Differences also reduce informal interaction, which may lead to a lack of task awareness and reduce trust among GDAD teams and team members (Holmstrom et al. 2006b; Lee & Yong 2010; Sharp et al. 2012). This will limit the efficiency and effectiveness of communication between teams or team members (Bosch & Bosch-Sijtsema 2010).

3.3.2.4.2 C2-Team Configuration

This category refers to the team size, number of teams, and coordination (e.g., early communication between teams, shared understanding, and cross-team communication) between GDAD teams (Agerfalk et al. 2009). This category has received a fair amount of attention in the selected studies and is mentioned in ten studies, corresponding to 48% of the selected studies. Paasivaara et al. (2013) and Green, Mazzuchi & Sarkani (2010) argue that it is important to pay more attention to early communication (i.e., at the beginning of the new project) among GDAD teams when implementing GDAD. Maruping (2010) argues that team member configuration (i.e., the management of team members across physical locations, such as 3-3-3 or 1-2-6) plays an important role in teamwork efficiency. The selected studies mention that GDAD teams or team members suffer from decreased communication speed and pair programming when a greater number of team members and teams are included in a GDAD project (Paasivaara et al. 2013). It has been reported that some members do not wish to communicate and have less understanding of team coordination (Dorairaj et al. 2011).

3.3.2.4.3 C3-Customer Communication

This category refers to communication with customers or with customer representatives (i.e., those who set the requirements or change the requirements of a project). This category has not received enough attention, comprising only 19% of the selected studies. In agile development, customers or customer representatives have to be

involved in the development process, and the project information must not be hidden from them (Korkala & Abrahamsson 2007; Korkala et al. 2010). However, in a GDAD environment, customer communication may be difficult. The selected studies have reported that a lack of customer communication may result in weak relationships and a misunderstanding of requirements, which may lead developers to either base decision on their experience or guess at the customer's requirements (Korkala et al. 2010; Layman et al. 2006).

3.3.2.4.4 C4-Project Characteristics

This category refers to such project characteristics as the project architecture (definition of a system and software structure), project domain, and type of project, such as single customer or commercial use (Agerfalk et al. 2009; Dingsøyr et al. 2014). This category has rarely been mentioned in the literature; it is mentioned in only two studies, representing 10% of the selected studies. Most of the findings have indirectly identified Project Characteristics as a challenge to GDAD communication. The challenges related to Project Characteristics are related to misunderstandings or an unnecessary communication flow due to the definition of a system and software structure and due to the definition of architecture used (Bosch & Bosch-Sijtsema 2010). A lack of predefined Project Characteristics decreases the knowledge sharing and communication capability of GDAD teams and team members, increases unnecessary communication, reduces integration with other processes, and decreases communication opportunities (Bosch & Bosch-Sijtsema 2010).

3.3.2.4.5 C5-Organisational Factors

This category refers to the tools and infrastructure capabilities that support GDAD communication, project management processes, and organisational culture. Project management processes refer to the processes (i.e., who is responsible for delivering what to whom and when) used to establish and maintain communication between GDAD teams and team members. Organisational culture is defined as the values, attitudes, and behaviours that represent an organisation's working environment and vision (Hofstede et al. 2010).

This category has attracted significant attention in the selected studies as a challenge for GDAD communication and collaboration. This category makes up the second highest percentage (i.e., 52%) of the selected studies. The tools used for GDAD communication include synchronous tools (e.g., phone, instant messaging) and asynchronous tools (e.g., email) [Holmstrom et al. 2006b]. The following issues related to organisational factors have been identified and reported in this study: the use of unsuitable tools by an agile team, technical incompatibilities between different sites (Paasivaara et al. 2013), the overhead of inadequate quality of video conference communication and coordination, and a lack of electronic tool support (Bosch & Bosch-Sijtsema 2010; Vallon et al. 2013). Gill and Bunker (2013) identified 14 categories (issues) that need to be taken into account when developing a comprehensive tool for GDAD communication. These categories include communication channel, business value, technology use case, quality, type, interface management, mode, access control, semantic interoperability, constraint, risk, contingency and disaster recovery, dependency and recommendation. The most appropriate tool can be chosen by using these 14 categories as a guideline for a particular GDAD communication context. Moreover, the bureaucratic organisation culture is considered a challenge in GDAD because it decreases the efficiency and effectiveness of communication among team members (Korkala et al. 2010).

3.3.2.4.6 C6-Human Factors

This category refers to differences in the language, national culture, trust, and personal practices between GDAD teams and team members. Language refers to the use of different languages among GDAD teams and team members. National culture refers to differences in national or local culture among GDAD teams and team members.

National culture is defined as the collective programming of the mind that distinguishes the members of one group or category of people from another. It encompasses norms, values, spoken language and styles of communication (Hofstede et al. 2010).

Trust refers to the difference in trust levels expressed by GDAD teams and developers depending on their location or country (Boden et al. 2007). Personal practice refers to

the differences in the personal attitudes and skills of GDAD developers (Holmstrom et al. 2006b).

This category is mentioned in 48% of the selected studies. A number of issues related to human factors have been identified and reported in this study: language misunderstandings and poor mutual understanding (Holmstrom et al. 2006b; Paasivaara et al. 2009), longer teleconference meetings, longer daily Scrum, the silence of some participants (Boden et al. 2007; Dorairaj et al. 2011; Paasivaara et al. 2009), confusion among team members (Holmstrom et al. 2006b), reduced coordination due to the observance of different holidays among the GDAD teams and personal (or group) attitudes (Bannerman et al. 2012; Bosch & Bosch-Sijtsema 2010), and different interpretations of the negative and sensitive issues of the project (Dorairaj et al. 2011; Holmstrom et al. 2006b).

3.3.2.5 Solutions for GDAD Communication Challenges

In this section, the techniques (strategies, tools, agile practices) used to address the GDAD communication challenges were organised into the same concept categories as for the GDAD communication challenges (Table 3.11). These recommendations were extracted from the same studies for each category. Table 3.14 summarises these recommendations.

3.3.2.5.1 C1-Distance Differences

The review identified a number of techniques to address issues related to Distance Differences: synchronise the work hours between the GDAD team members, synchronise communication between different teams or between the local Product Owner and distributed Scrum Master, scale Scrum with the use of Scrum of Scrums, divide the meeting into several parts (distributed and onsite parts), create local teams for each geographical or same time-zone area, hold strict all-hands meetings in which all members are required to attend or share, reduce the number of whole distributed project meetings, use asynchronous tools (e.g., Wiki, email, and so on), minimise dependencies among teams, combine the flexibility of agile methods with the rigidity of traditional methods, enhance regular visits and face-to-face communication, split the project into

small parts, and centralise the experts in the home country (Bannerman et al. 2012; Boden et al. 2007; Green et al. 2010; Layman et al. 2006; Lee & Yong 2010; Paasivaara et al. 2009; Sharp et al. 2012; Sindhgatta et al. 2011; Zieris & Salinger 2013).

Table 3.14: Techniques to overcome GDAD communication challenges

Ref.	Category	Techniques
C1	Distance Differences	• Strategies: small difference in time-zone countries to be chosen for distribution, divide work between no more than 2 sites, divide meeting to several parts (distributed and onsite parts), provide more flexibility in working environment, combine the flexibility of agile methods with the rigidity of traditional methods, encourage regular visits and face-to-face communication, create structure of trust, keep dependencies as low as possible among teams, encourage individual to work closely with both developers and project management teams, enforce meetings and commitment, localize component ownership, enhance coordination by promoting social skills • Tools: switch for the most appropriate tools, use synchronous and asynchronous tools, use architectural roadmap, use secure shared information repositories • Agile practices: use information hub such as Product Owner, synchronise communication between local Product Owner and distributed Scrum Master, scale Scrum with the use of Scrum of Scrums
C2	Team Configuration	• Strategies: encourage face-to-face meeting at the beginning of each project, promote product demonstration at the end of each iteration to ensure communication synchronisation, encourage periodic meetings, use simple design, form local teams, promote clear roles and responsibilities for each site, systematically check for refactoring, involve customers in development, encourage trusted relationship between distributed teams and team members, increase effective formal communication • Tools: use synchronous communication tools • Agile practices: avoid distributed pair programming, encourage small release
C3	Customer Communication	• Strategies: ensure regular agile meeting and customer should be involved, promote the customer's representative who plays the role of the customer up front • Agile practices: promote frequent interaction with Product Owner (Scrum roles)
C4	Project Characteristics	• Strategies: get agreement on the requirements by all teams and relevant members at the beginning of the project, promote coordination with the overall strategy and local process, offshore representative should participate in onshore daily meetings to help resolve any misunderstanding • Tools: use explanation of reference architecture (overall architecture) by all teams and relevant members
C5	Organisational Factors	• Strategies: promote corporate organisational culture that supports rapid communication, trust between agile stakeholders, and obtaining quick feedback from the customers • Tools: use different communication technologies, use available project management tools • Agile practices: employ Scrum Master and Product Owner for each site, encourage frequent visits of Scrum roles to other sites, encourage joint daily Scrum/ 2week review and monthly retrospective meetings, employ

		single-site Scrum teams only
C6	Human Factors	• Strategies: exchange visits between distributed sites, all developers should be seen equal, encourage connection team leader, speak slowly and clearly, ensure awareness of the members about meeting's matters before meeting, address the language issue as early as possible in the project, encourage individual to work closely with both developers and project management teams, use less detailed communication, use easily edit Wiki pages, keep shared information updated • Tools: use tools to record meetings, use asynchronous communication technology • Agile practices: Scrum practices, use local Scrum Master instead of Product Owner to answer questions, encourage offshore representative participates in onshore daily Scrum meetings, synchronise the Sprint length

3.3.2.5.2 C2-Team Configuration

A number of techniques to address the Team Configuration issues were identified in this review: frequently demonstrating products, promoting face-to-face meetings at the beginning of each project (Paasivaara et al. 2013), promoting periodic meetings between different teams, and using synchronous communication (telephone, video, etc.). Moreover, other authors recommended using a simple design and promoting small releases, creating local teams and using clear roles and responsibilities for each site, avoiding distributed pair programming, and systematically checking for refactoring (Maruping 2010; Valimaki & Kaariainen 2008). In addition, it was recommended that more attention should be paid to the importance of the role of the Product Owner among GDAD teams and inside a co-located team (Paasivaara et al. 2013). Others suggested increasing effective formal communication (e.g., inception meetings at the beginning of the project, weekly or daily meetings with other GDAD members, and meetings with customers) and encouraging mutual trust among team members (Dorairaj et al. 2011).

3.3.2.5.3 C3-Customer Communication

This review identified a number of techniques for addressing issues related to customer communication. These techniques included using a customer representative to play the role of the customer (Korkala & Abrahamsson 2007; Layman et al. 2006), enhancing rapid communication, promoting regular agile meetings and active customer engagement (Korkala et al. 2010), and promoting the role of Product Owner (Scrum roles) [Paasivaara et al. 2013].

3.3.2.5.4 C4-Project Characteristics

This review also identified some techniques for addressing project characteristics issues. These techniques include using an overall architecture view among GDAD teams, coordinating the overall strategy with the local team's process (Bosch & Bosch-Sijtsema 2010), and encouraging offshore representatives to participate in onshore daily meetings to help resolve any misunderstandings (Bannerman et al. 2012). Other suggestions include using reference architecture and ensuring agreement on the project's requirements by all teams and relevant members at the beginning of the project (Bosch & Bosch-Sijtsema 2010).

3.3.2.5.5 C5-Organisational Factors

This review also identified a number of techniques for addressing Organisational Factors issues: offering different communication tools, using available project management tools, using different communication models (Dorairaj et al. 2011; Layman et al. 2006; Zieris & Salinger 2013), employing Scrum Master and Product Owner for each site, and enabling frequent visits of Scrum roles to other sites (Yadav et al. 2007). In addition, use of the communication technology assessment tool (CTAT) to assess and select an appropriate tool for supporting GDAD communication was suggested (Gill & Bunker 2013). Moreover, a corporate organisational culture that supports rapid communication and trust between agile stakeholders was considered a success factor for GDAD (Korkala et al. 2010; Layman et al. 2006).

3.3.2.5.6 C6-Human Factors

The authors in this review have recommended some techniques to address Human Factors problem: synchronising the Sprint length, encouraging offshore representatives to participate in onshore daily Scrum meetings (Bannerman et al. 2012), using Scrum of Scrum practices (Paasivaara et al. 2013), speaking slowly and clearly, informing team members about a meeting's topics before the meeting, addressing the language issue as early as possible in the project, using less detailed communication (Dorairaj et al. 2011), keeping shared information up-to-date, using a local Scrum Master instead of Product Owner to answer questions in the case of large teams (Lee & Yong 2010), enhancing trust in relationships and shared understanding through exchanging visits (Holmstrom et

al. 2006b; Layman et al. 2006; Paasivaara et al. 2009), and using multiple communication modes and tools to record meetings (Holmstrom et al. 2006b; Lee & Yong 2010).

3.3.2.6 Impact of Communication on GDAD Success

Communication efficiency and effectiveness have been reported to play a critical role in the success of GDAD projects or releases (Boden et al. 2007; Bostrom 1989; Dorairaj et al. 2011; Korkala et al. 2010; Layman et al. 2006). While communication is crucial, communication technologies should also be an integral part of the GDAD environment to facilitate communication and thereby successful GDAD projects (Dorairaj et al. 2011; Gill & Bunker 2013; Korkala & Maurer 2014; Yadav et al. 2007). There are a number of communication tools to choose from (e.g., Lync, Skype). However, using a combination of communication tools that fit the GDAD context is considered more appropriate than using one single communication tool because tools differ in scope (Gill & Bunker 2013; Green et al. 2010). Effective customer communication should be considered a key factor for successful GDAD projects (Korkala & Abrahamsson 2007; Korkala & Maurer 2014). It is important for a successful GDAD project to have efficient and effective knowledge sharing and transfer, which allow team members to understand the customer requirements and help team members plan and perform development activities (Dorairaj et al. 2011; Korkala et al. 2010). Poor communication can cause severe problems for GDAD projects, including leading to inferior-quality products and delayed software delivery (Korkala & Abrahamsson 2007; Layman et al. 2006). More-over, agile practices in GDAD are useful for enhancing communication and product quality (Holmstrom et al. 2006b; Paasivaara et al. 2009).

3.3.2.7 GDAD Communication Patterns

The meta-analysis revealed that some GDAD communication patterns have been identified in the selected studies. These patterns include strategies, tools, and agile practices (e.g., Scrum and XP). In the strategy patterns, frequent deliveries and systematic coordination to the small release were recommended as successful GDAD asynchronous communication loops that can serve as a sufficient surrogate for synchronous communication (Layman et al. 2006; Maruping 2010).

Moreover, synchronising work hours, reducing the number of GDAD team meetings, hosting team gatherings at the beginning of each project, exchanging visits between GDAD teams, and maintaining key documentation were found to support GDAD communication (Bannerman et al. 2012; Bosch & Bosch-Sijtsema 2010; Holmstrom et al. 2006b; Lee & Yong 2010; Sharp et al. 2012). In addition, corporate culture is considered as a success factor for GDAD communication (Korkala et al. 2010).

In the tool patterns, access to appropriate means or tools of communication that match the situational needs of the user (e.g., developer) was found to be a key factor for enhancing GDAD communication (Boden et al. 2007; Gill & Bunker 2013; Green et al. 2010; Valimaki & Kaariainen 2008). Other successful GDAD communication tool patterns included using a central digital repository for the customer requirements and the assigned tasks for each agile developer (Gill & Bunker 2013; Maruping 2010; Valimaki & Kaariainen 2008). This repository is accessible by all team members and allows all team members to view and manipulate the progress made in satisfying the requirements in real-time (Maruping 2010).

In the agile practice patterns, team members play important roles in GDAD communication. For example, component leads and project managers have a significantly higher communication overhead than development team members (Sindhgatta et al. 2011). The coordination responsibilities of component leads and project managers make them the focal point of many communication links in GDAD. Thus, together with technical acumen, component leads and project managers need to possess strong social skills to enhance communication and collaboration across sites (Sindhgatta et al. 2011). They need to be able to travel when needed (Valimaki & Kaariainen 2008). Moreover, pair programming in GDAD was found to enhance communication among distributed team members (Sharp et al. 2012). Pair programming in GDAD can be achieved using screen sharing and audio calls (Sharp et al. 2012). GDAD Scrum practices offer a distinctive advantage in mitigating GDAD communication challenges (Bannerman et al. 2012; Valimaki & Kaariainen 2008). The following practices were found very effective for addressing poor communication in GDAD (Valimaki & Kaariainen 2008): the Product Owner should represent the

customer and interact with the development team, local Product Owners and Scrum Masters are needed for each site to facilitate communication, clear Product/ Sprint Backlog items should be specified in detail, and specifications should be able to contain rich information to facilitate understanding (e.g., using figures, diagrams) among team members.

3.4 GDAD Communication SLR Overview- State-of-the-Art

It is still arguable whether agile practices can be effectively scaled up and used in GDAD environments due to communication challenges (Batra et al. 2010; Hummel et al. 2015). The findings of this review are summarised in seven points. First, despite its acknowledged importance, we found that our knowledge about GDAD communication in practice is limited. This is the result of the study results being scattered, inconclusive, and ambiguous and scarcely any studies opening up the communication process or focusing on the social interaction and behaviour of teams as part of the research. It is not well-established how high communication efficiency and effectiveness can be achieved in this context. However, this SLR study confirms that there is a growing interest in GDAD communication, and most of the selected studies have reported the possibility of a successful implementation of GDAD and overcoming communication challenges (e.g., Dorairaj et al. 2011; Yadav et al. 2007).

Second, our findings reveal that Distance Differences have been the most reported challenges for GDAD communication. Additionally, Team Configuration, Organisational Factors and Human Factors have been reported to have significant negative impacts on GDAD communication. Other challenges have been rarely reported, but they may have a strong effect on GDAD communication. For example, Project Characteristics have been reported in only two studies despite software architecture being considered a substantial challenge in the traditional DSD environment.

Third, our findings also indicate that certain techniques should be used to support GDAD communication. GDAD communication requires some temporal overlap between GDAD teams or stakeholders to allow meetings to be held. Communication

time must also be reduced to avoid temporal overlap, and supported distributed strategies should be used. These strategies can be implemented by synchronising work hours, creating local teams, increasing the use of local meetings, using strict communication policies, reducing the number of GDAD team meetings, hosting team gatherings at the beginning of each project, exchanging visits between GDAD teams, requiring presentations by all team members, maintaining key documentation, and so on. Moreover, the current available technologies (e.g., Skype, IM) can be used to support and enhance GDAD communication.

Fourth, many studies have reported that using Scrum practices in GDAD may enhance communication and collaboration between GDAD teams (e.g., Paasivaara et al. 2009; Valimaki & Kaariainen 2008; Zieris & Salinger 2013). This is consistent with the fact that the Scrum is the most preferred agile method among GDAD teams. These studies have argued that Scrum practices enhance the GDAD communication through: (1) Sprint planning meetings, which can be arranged by teleconference or using exchange visits between teams, and dividing meetings into several parts (i.e., onsite and offsite), (2) daily Scrum meetings between distributed team members, (3) retrospective meetings, (4) a Scrum Master, who shapes the information flow between onsite and offsite teams, (5) a Product Owner, who can represent the customer, and (6) the use of a backlog that all distributed members can access. However, Valone et al. (2013) reported that the members of one Scrum team should not be distributed over several sites and that every site should have at least one Scrum Master and one Product Owner.

Fifth, it has been noted that although knowledge about how to manage traditional distributed software teams' communication has accumulated over the past few decades, the governance of GDAD teams' communication (e.g., using software architecture guidelines) has not been paid enough attention. Indeed, the incorporation of agile methods into traditional DSD environments poses many new challenges, highlighting the need for a better understanding of human and communication issues in the GDAD context (Fruhling & Vreede 2006). For example, it is not clear whether GDAD communication is characterised by a higher or lower intensity, frequency, or quality comparing to traditional DSD. None of the selected studies have attempted to compare or contrast communication in distributed traditional development to GDAD

communication. Evidence suggests that instead of abandoning traditional DSD project communication principles, one should take advantage of these principles and combine them with agile communication focused project management practices (Barlow et al. 2011; Batra et al. 2010; Dyba & Dingsoyr 2008; Hummel et al. 2015; Karlsson & Agerfalk 2009).

Sixth, structural properties of communication have long been recognised as important drivers of project performance and success in product development (Brown & Eisenhardt 1995; Cataldo & Ehrlich 2011; Gokpinar et al. 2010). While there are a few studies that examine communication structure in SD (e.g. Cataldo & Herbsleb 2008), there is little research, which has been extended to examine the specific relationship of communication's structural properties and GDAD outcomes such as productivity or quality (Cataldo & Ehrlich 2011; Inayat & Salim 2014).

Finally, we found that there is an abundance of qualitative and exploratory studies but not confirmatory and explanatory studies. Similarly, design-oriented studies and action-oriented research using the available communication tools could help clarify the effects of the use of different GDAD communication practices. Furthermore, to obtain a deeper understanding of GDAD communication, previous authors have recommended that the research design needs to fit the current state of theory and research (Dingsøyr et al. 2014; Dyba & Dingsoyr 2008). Therefore, the role of communication in GDAD could be investigated from the perspectives of related theories, such as Common Ground (Clark & Brennan 1991), Activity Theory (Engeström 1999), and Coordination Theory (Malone & Crowston 1990). All such studies would increase the knowledge about GDAD communication and its impact on project success.

3.5 Thesis Motivation and Literature Gaps Fine-Tunning

This chapter has reviewed the current literature on GDAD communication challenges, the techniques used to mitigate the effect of these challenges, and the impact of communication (i.e. efficiency and effectiveness) on GDAD project's performance. The findings of the SLR ensure and clarify the research gaps that were identified in Ch.2, which relates to theory and practice of GDAD communication. First, the literature lacks

the theoretical glue or the conceptual foundations (Abrahamsson et al. 2009; Lee & Xia 2010). This research will fill up this gap by employing and building a conceptual research model based on Common Ground theory (Clarke & Brennan 1991). Common Ground theory will be discussed in more detail in Ch.4.

Second, the literature lacks empirical (i.e. confirmatory and explanatory) studies to understand GDAD communication efficiency and effectiveness. Moreover, empirical evidence regarding the Project Characteristics (i.e. Project Architecture and Project Domain) challenges is very scarce despite the common agreement among experts that GDAD should be paired with some of the distributed plan-driven principles to succeed such as using structured communication and documentation (e.g., Batra et al. 2011; Fruhling & De Vreede 2006; Hoda et al. 2010; Ramesh et al. 2012; Stettina et al. 2012). Structured communication (e.g., using agile EA elements shared) provides the information needed by all distributed teams to get their work done without incurring a heavy overhead (Cataldo & Ehrlich 2011; Ramesh et al. 2012). Hence, to begin filling up this gap, a research model is developed (see Ch.4) that employs agile EA as a means of communication and a technique to mitigate the impact of communication challenges in GDAD environment. This model is developed based on the available literature and the framework developed in this SLR (Figure 3.2). This model (research model) will be discussed in the next chapter (Ch. 4), in full detail.

3.6 Rationale for Adopting EA in GDAD

As was found from the SLR (C4-Project Characteristics), using architecture was found among the GDAD communication challenges although the number of studies that have mentioned it was very scarce. Moreover, as it was discussed in Section 2.7, there is a common agreement among the practitioners and academics that pure agile methods in distributed environment may be not doable and that GDAD needs to use some of plan-driven principles (e.g., Batra et al. 2011; Hummel et al. 2015; Ramesh et al. 2012).

It is important to clarify that using EA in this context refers to using the artefacts (elements) of EA at a minimum (just enough) level; examples of these artefacts are tables, diagrams, description, and so on. In other words, traditional EA approaches are

considered too heavy for agile development. A light-weight agile EA is required for GDAD, which seems more suitable to the people and active communication-driven agile development ways of working than the traditional documentation-driven and heavy process-centric EA approach (Gill 2015a). That is, for example, the development team does not spend a long time on designing and developing the architecture for current and future requirements; rather, only the architecture infrastructure the project needs for the current requirements is designed and developed, which keeps the system simple, increases developments speed, and reduces overhead (Fruhling & De Vreede 2006; Martin 2003).

Moreover, agile EA offers a holistic and evolving shared view of the integrated information of business and IT architecture domains to enable effective and efficient communication among GDAD stakeholders. Usually, development teams rely on isolated software or IT architecture. EA as a holistic and integrated business and IT information will ensure that the important points of the whole EA are not overlooked by the GDAD teams. EA is perceived to be a glue to keep the GDAD teams aligned towards a shared vision (Edwards 2006).

In addition, this study argues that this holistic integration view of agile EA will help overcome most GDAD communication challenges, since it will decrease the frequency of communication, decrease exchange visits and training, increase coordination, and increase the common understanding of the software requirement between GDAD teams. Also, this study argues that using agile EA shared view will have a positive impact on the software performance delivered in GDAD. This view and the reasons for choosing agile EA as a means for GDAD communication will be discussed in detail in next chapter (Ch.4).

3.7 Chapter Summary

Communication has been widely recognised as the main challenge of GDAD. However, no study has systematically reviewed and synthesised the empirical studies on GDAD communication. This chapter presented a SLR with a view to identify the communication challenges that need to be addressed for establishing efficient and

effective GDAD communication. The identified challenges of GDAD communication were categorised into six key categories: Distance Differences, Team Configuration, Customer Communication, Project Characteristics, Organisational Factors, and Human Factors. Each category includes some communication challenges. The Distance Differences category was the most frequently mentioned category in the literature. The most rarely mentioned category was Project Characteristics. The negative impacts of GDAD communication challenges as well as the techniques for mitigating those impacts were also identified. In addition, the impact of communication on GDAD success (performance) was identified, with a positive effect of communication efficiency and effectiveness on GDAD success being supported by the literature.

This review enabled the researcher to construct the current state-of-the-art about GDAD communication challenges. It also helped to identify some gaps in the literature, which helped the researcher to fine-tune the research (thesis) problem and questions. Moreover, a big need was identified in the literature to empirically study how GDAD communication can be enhanced by employing traditional DSD principles such as indirect communication using EA artefacts, and how GDAD communication can enhance GDAD performance. Drawing on the background discussion, a conceptual model will be proposed in the next chapter that aims to address the research gaps identified in the previous two chapters. The research model constructs; agile EA, active communication, and GDAD performance, as well as the relevant propositions between constructs relationships will be discussed.

Chapter 4 Research Model

In Ch.2, GDAD communication literature was reviewed and some gaps were identified. Then, in Ch.3 the latest empirical published work regarding GDAD communication was systematically reviewed, which helped to fine-tune the literature gaps and the scope of this study. The review revealed that agile EA is one of the potential techniques/ strategies that would enhance GDAD communication and GDAD software performance. In addition, it showed that the research on GDAD communication is not only scant in terms of quantity, but also lacks both theoretical glue (Conboy 2009, p. 330) and empirical evidence.

4.1 Introduction

Having clearly identified the scope of this study, now we can move forward in reviewing the literature on the research model constructs. As mentioned in Ch.3, the research model is a part (special case) of the GDAD communication framework (Figure 3.2), which was built based on the empirical literature and theoretically glued concepts using Common Ground theory and The Unified Model of Information Software Development Success. The research model will empirically investigate agile EA as a communication enabler and input to the active communication process which includes both communication efficiency and communication effectiveness (dimensions of active communication). Both agile EA and communication impacts will be tested on the performance (success) of the software developed in GDAD environment. The three research model's constructs (i.e. agile EA, active communication, software performance) as well as Common Ground theory will be discussed in detail in this chapter. Moreover, the hypotheses built between the three constructs will be discussed.

Agile EA can increase the common ground, which minimises the communication and increases coordination among GDAD teams and members. Agile EA can be used to

identify the communication needs in coordination requirements, which results in faster task development accomplishment (Cataldo et al. 2008).

The structure of the chapter is as follows. In Section 4.2, the theoretical foundation (i.e. Common Ground theory) of the research model is discussed. In Section 4.3, agile EA is discussed. This section discusses an overview of traditional EA first, and then agile EA is discussed in detail. Section 4.4 discusses the study proposition of employing agile EA and how it is going to mitigate the impact of GDAD communication challenges. Each of the constructs that made up the model, their definitions as well as the hypothesised relationships that link them with one another will be discussed in Section 4.5 and Section 4.6. Finally, Section 4.7 provides a brief summary to this chapter.

4.2 Common Ground Theory

Although GDAD has grown in recent years, there is a lack of empirical evidence for its effective use and predicted improvements in software development. One of the main challenges related to this branch of research (as discussed in Ch.2 and Ch.3) is the lack of sound conceptual foundations and theoretically oriented empirical research into the use of agile methods (Abrahamsson et al. 2009; Agerfalk et al. 2009). Theories such as Media Synchronicity Theory (Korkala & Maurer 2014) and Control Theory (Maruping et al. 2009) have recently provoked interest in the agile research community. To address this theoretical oriented gap, the research model of this study is based on Common Ground theory (Clarke & Brennan 1991). The purpose is to enhance GDAD communication. The focus is on how to decrease the frequency of communication (Batra et al. 2010; Ramesh, Mohan & Cao 2012), and increase the efficiency and effectiveness of communication in GDAD environment (Holmström et al. 2006b; Pikkarainen et al. 2008), at the same time, which will enhance the GDAD performance (Balijepally et al. 2009; Drury-Grogan 2014; Sawyer et al. 2010).

Clark defines common ground between two people as "the sum of their mutual, common or joint knowledge, beliefs and suppositions" (Clark 1996, p. 93). The notion of common ground has been developed (Clarke & Brennan 1991) to explain language usage as a coordination game, and since then it has been used to study communication

and coordination in organisations (Puranam et al. 2009). Common ground had its origin in the fields of linguistics and cognitive psychology but has now been applied in other domains such as Human Computer Interaction (Monk 2003). Language is viewed as a collaborative activity that uses existing common ground to develop further common ground and hence to communicate efficiently (Monk 2003).

According to Clarke (1996), common ground can be characterised in terms of three basic categories: initial common ground, public events, and the current state of the work. Initial common ground includes all the relevant knowledge and prior history about the work. Firstly, initial common ground involves not only the shared general knowledge about the work and environment, but also all the conventions that are associated with their work or particular task (Klein & Bernstein 2004). It also involves what team members know about each other prior to working together (Klein & Bernstein 2004). For example, a development team conducts a particular part of a project or system and another distributed team conducts another part of the project with different culture setting (different distributed teams can have different communication procedures or different development design).

Secondly, public events involve knowledge of the event history (i.e. the activity the participants have engaged in up to the present point in doing a work) (Klein & Bernstein 2004). For instance, it is important to know that options may have been lost irretrievably due to time or resource dependence. Once everyone is informed about the relevant aspects of previous history, the team members' ongoing work together provides them with further information about each other (e.g., culture and skills), establishes patterns regarding how certain situations have been handled, and helps new members to be included efficiently. Finally, the current state of the work provides hints that enable prediction of subsequent actions and the formulation of appropriate forms of communication. However, team members should not be misled by such situations that may frame falsely thinking that they are viewing the same thing and interpreting it the same way (Klein & Bernstein 2004).

In general, common ground is what makes coordination work (Klein & Bernstein 2004). It enables coordination because it allows team members who possess similar knowledge

to accurately anticipate and interpret each other's actions in interdependent tasks or the meaning implied by certain communication means (e.g., words) (Puranam et al. 2009). Most of the important types of mutual knowledge, beliefs, and assumptions were found about the roles and functions of each team member. This involves the tasks that the team is capable of executing, the skills of each member, the goals of the members (including their commitment to the success of the team project), and the "stance" of each member (e.g., his or her perception of time pressure, level of exhaustion, and competing priorities) (Klein & Bernstein 2004). Moreover, the degree of quality of common ground can vary due to the particulars of people, circumstances, and their current objectives (Klein & Bernstein 2004).

Common ground is not only a state of having the same knowledge, data, and goals, but rather a process of communicating, testing, updating, tailoring, and repairing mutual understandings (Brennan 1998; Vlaar et al. 2008). It is a continuous achievement rooted in communication and interaction (O'Leary et al. 2014; Vlaar et al. 2008). It can help us understand the way that technology may affect the process of communication (Monk 2003; Wilson et al. 2013). As team members collaborate, they realise that they belong to the same social category, and they develop and use a common pool of joint actions and patterns (Clark 1996).

Perhaps the most important basis for interpredictability (Klein & Bernstein 2004) is common ground (Clark & Brennan 1991), that support interdependent actions in communication process between two parties where successful grounding in communication requires "to coordinate both the content and process" (Clark & Brennan 1991; Cramton 2002). Effective communication between people requires that the communicative exchange take place with respect to some level of common ground (Clark 1996; Olson & Olson 2000). When common ground is high and accurate, communication is more likely to be understood as intended (Cramton 2002; Vlaar et al. 2008). The notion of efficiency was reformulated by Clark and Brennan (1991) as a matter of minimising communication costs and then used to predict the effects of different ways of mediating communication (Monk 2003). Moreover, not only can common ground enable coordination primarily through the use of formal mechanisms (e.g., software architecture, documentation, and procedures), but also give rise to tacit or

informal communication and coordination (Puranam et al. 2009). This will only be the case when there is sufficient common ground across team members, which enables successful communication and allows members to adjust their actions appropriately to each other (Puranam et al. 2009). In addition, with common ground, actions (e.g., tasks) are aligned not because communicating members are mandated to take aligned actions, but because they share sufficient knowledge that enables active communication and alignment of their actions (Puranam et al. 2009). In this sense, informal communication based on common ground can replace (to a large extent) formal communication, which is subjected to less disruption effects than that accompanies formal form, because no considerable changes to the formal organisation are necessary (Puranam et al. 2009). In other words, the need for formal communication should decline in the presence of common ground, as it can at least partly substitute the effects of formal communication by resolving other kinds of communication and coordination problems (Puranam et al. 2009).

In co-located ASD environments establishing and negotiating common ground is a natural process, because the shared location provides the shared context whereby team members can be part of each other's workspace and share artefacts related to a project easily (Modi et al. 2013). In GDAD environment, the common ground cues are reduced due to the distance, and therefore, team members can face a lack of active communication and generally experience more challenges in maintaining active communication needed for the progress of their tasks, the artefacts and their respective offshore team members (Herbsleb 2007; Hinds & Mortensen 2005).

However, in GDAD contexts, active communication can take place, but the effort is usually quite large (O'Leary & Mortensen 2010; Olson & Olson 2000; Polzer et al. 2006). Once GDAD team members share the sense of belonging to the same category and develop a set of common tasks or projects, they will be open to communicate more and share more knowledge with the objectively distant members as they feel they are more proximate (O'Leary et al. 2014; Rivera et al. 2010). The change of communication medium from face-to-face to written form (e.g., email) in GDAD environment permits re-reading the message, which helps each party to ensure that they really achieve common ground (Monk 2003). Moreover, common ground allows GDAD team's

members to use abbreviated forms of communication with less frequency and still be confident that potentially ambiguous messages and signals will be understood (Klein & Bernstein 2004). Clark and Brennan (1991) use the term "grounding" to describe the process of developing the mutual shared understanding which allows team members to establish and negotiate common ground in an incremental manner. Grounding then is applicable for the communication process in the GDAD environment, where teams continuously undergo the process of constructing and re-constructing a mutual shared understanding regarding a joint tasks or projects, thus establishing and negotiating common ground through a process of realignment (Cramton 2002; Olson & Olson 2000).

According to Calefato et al. (2012) when distributed groups interact only via text-based communications, greater common ground can be achieved but it takes much longer to achieve than face-to-face interactions. Common ground was identified as one of the essential pre-requisites concepts of successful communication and collaboration in DSD environment (Olson & Olson 2000). In other words, greater common ground prevails as vital scaffolding to communication and collaboration in GDAD environment since it allows GDAD teams to achieve shared joint knowledge clearly and easily. When common ground is weakened, it can cause problems with tightly coupled work activities where team members are dependent on others for the progression of project tasks due to the poor communication (less efficient and effective communication).

In the globally GDAD teams' settings, distributed teams have diverse cultural backgrounds and different spoken languages, which can hinder the grounding process (Cummings 2004; Hinds & Mortensen 2005). Consequently, different assumptions, priorities, and sufficient common ground have to be developed to carry out the joint communication and collaborative activity. Failure to establish and maintain common ground can result in serious breakdowns in the communication and the collaborative work (Cramton 2002). Clark and Brennan (1991) advocate eight "constraints" that affect the grounding process: (1) copresence, (2) visibility, (3) audibility, (4) cotemporality, (5) simultaneity, (6) sequentiality, (7) reviewability, and (8) revisability. The greater the constraints a medium can address the greater common ground and the better active (efficient and effective) communication process can be achieved (Clark &

Brennan 1991; Modi et al. 2013). One of the key principles of grounding is where team members use the best available medium that leads to the least communication and collaborative effort (i.e. more efficient communication and collaboration) (Monk 2003).

Several studies have suggested potential requirements, design and development of specific systems which could be used in a globally DSD context to increase the common ground (e.g., Froehlich & Dourish 2004; Storey et al. 2005). In practice, however, where distributed teams have adapted agile methods they usually use existing tools and systems grouping them to produce hybrid structures to support awareness cues needed within the teams through the use of synchronous and asynchronous processes.

Prikladnicki et al. (2012) examined how the lack of awareness related to work items can cause communication breakdowns within global software teams. Unfortunately, at present there is not one single independent awareness support tool which encapsulates all the requirements needed for GDAD teams; organisations, therefore, have to prioritise their collaboration needs and their tools to support them (Prikladnicki et al. 2012).

Dabbish et al. (2012) reported how developers in an open source environment valued the visibility and transparency of collaborative platforms such as GitHub where the shared artefacts are visible to all the participants. Smite and Dingsøyr (2012) studied awareness challenges in three DSD teams where elements of Scrum methods were being applied. They reported that specific awareness systems discussed in the literature were still not widely used in practice and there were multiple challenges in spreading awareness in such settings.

Although some researchers reported that GDAD can be successful using pure agile principles, and that there is no need to employ any principle of plan-driven DSD such as using EA (e.g., Paasivaara & Lassenius 2014; Sutherland et al. 2009), others found that GDAD ultimately needs to be injected with some plan-driven principles such as EA to achieve equivalent software performance as co-located agile development (e.g., Bosch & Bosch-Sijtsema 2010; Kornstädt & Sauer 2007; Kuusinen et al. 2012). We believe that these conflicting findings may be due partly to the projects sizes and domains, and the team member's cultural categories differences in both cases; however, including

many distributed teams from different cultural backgrounds may require some tailoring to the agile development principles in order for GDAD projects to be successful (Batra et al. 2011; Maruping et al. 2009). Hence, we believe that more emphasis should be given to common ground (e.g., using agile EA elements) that will enhance GDAD communication process and performance (see Sections 4.4, 4.6.1, and 4.6.2). Once GDAD team members share a common knowledge and perceive each other as proximate, their communication will be more efficient and effective, they will be more open to share experience and learning from each other, and they will be more willing to work together again in the future.

4.3 Agile Enterprise Architecture

In this section, a detailed discussion is provided about agile EA and an overview of some agile EA frameworks. However, before that an overview about the traditional EA definitions and the most common EA frameworks will be provided.

4.3.1 Traditional Enterprise Architecture Overview

Many definitions have been provided in the literature to EA. EA is defined as "the organising logic for business processes and IT infrastructure, reflecting the integration and standardisation requirements of the company's operating model" (Ross et al. 2006, P.9). EA is also defined as "the definition and representation of a high-level view of an enterprise's business processes and IT systems, their interrelationships, and the extent to which these processes and systems are shared by different parts of the enterprise" (Tamm et al. 2011, p. 147). EA is also defined as "a comprehensive approach for business development. In its essence, it involves the alignment of organisations' business capabilities, information and information technology (IT) to a common goal through the production and utilisation of architectural models" (Niemi & Pekkola 2015, p. 1). Another way of looking at EA is as a means to inform, guide, direct, and constrain the decisions taken by human beings within organisations (Janssen 2011).

According to Gill (2013a), enterprise and architecture are two terms to be defined. Firstly, enterprise is defined as "... an entity, regardless of its legal form... including partnerships or associations regularly engaged in economic activities" (European Union

Commission 2003) such that enterprises have to identify the needed capabilities of the smoothness and adaptation processes (Gill 2013a). Secondly, architecture is defined as the "fundamental concepts or properties of a system in its environment embodied in its elements, relationships, and in the principles of its design and evolution" (ISO/IEC/IEEE42010 2007). Accordingly, Gill (2013a) define EA as "a blueprint that describes the overall structural, behavioural, social, technological, and facility elements of an enterprise's operating environment that share common goals and principles" (Gill 2013a, p.1).

The fundamental tools and methodologies are required to design and manage EA through their lifecycles. These tools are based on fundamental principles which are referred to as an architecture framework. Architecture framework is defined as conventions, principles, and practices for the description of architectures established within a specific domain of application or/and community of stakeholders (ISO/IEC/IEEE42010 2011).

ISO/IEC/IEEE 42010 defines the requirements of EA as follows:

1. Architecture description: it is an artefact that describes the architecture of the system (project) of interest (ISO/IEC/IEEE42010 2007). It describes how to document the architecture from different views that include the description for entities and relationships, which takes the form of tables, lists, diagrams, or any other high composite level (ISO/IEC/IEEE42010 2011). Architecture description is expected to includes (ISO/IEC/IEEE42010 2011):

    1. identification of the project stakeholders,
    2. identification of stakeholders' concerns,
    3. definitions for different viewpoints that form these concerns,
    4. correspondence rules that integrate these viewpoints,
    5. business value measurement matrices,
    6. EA initiatives and maturity models,
    7. enterprise communication model, and
    8. architecture's rationale (explanation and justification of the EA).

2. Architecture viewpoint: it addresses the stakeholder's concerns of the project and describes the architecture of the project in accordance with rules and conventions that are defined in the viewpoint. View point formalises the idea of the existence of different views of the same project. On the other hand, it plays an integral part of architecture framework and description. A view point needs to include: (1) stakeholder's identification, (2) stakeholder's concerns, (3) the used model (i.e. means that represent the relationships such as N-Squared), and (4) conventions, techniques, notations and operations used in the models.

3. Architecture framework: defines how to build/create and use EA. It provides principles, practices, polices, and standards for the description of architecture. It offers tools and approaches to manage the complexity of the context. These tools and approaches help architects to abstract from the detailed description to get focused design tasks and produce an architecture description document. Architecture framework needs to include: (1) relevant stakeholders, (2) stakeholder's concerns, (3) viewpoints, (4) rules integrating these viewpoints, and (5) processes, methods, tools, and other needed practices.

4. Methods for designing architecture: the processes that are used or followed by the architect in designing the architecture such as overarching EA process, breaking it into lower or smaller activities, and phases composition. A process is defined by its input, output, objectives and steps or activities, and may be supported by tools, principles, techniques, practices and rules.

5. Organisation of architects: it provides guidance to the team's structure, skills, experience, governance and training.

6. Layers of EA: each layer contains components that execute processes and offer services to the upper layer. The upper layer delegates work to the lower layer. Ideal EA framework has the following layers: (1) external/outer layer (environment) that represents the external entities and activities which are directed, supported and

monitored by the business, (2) business layer that represents the business functions, which offer services to each other and to the outer layer entities, (3) data layer that represents the business information and other stored data, (4) information system layer that represents the business applications which offer information services to each other and to business layer entities, and (5) technology layer that represents hardware, network, platform and tools applications that offer platform services to each other and to information system layer entities.

Therefore, EA can be seen through the disciplines and principles of an organisation. EA represents a roadmap of an organisation through information, business, and process and technology changes necessary to execute their strategies. EA is essential for enterprise development and it is the base for the continuous enterprise adaptation practices (Gill 2013a). Moreover, EA provides a long-term view of an organisation's processes, technologies, and systems, which enable individual projects to build capabilities rather than just fulfil immediate needs (Lange et al. 2015; Niemi & Pekkola 2015).

There is a number of well-known architecture frameworks that have been developed during the last years such as Zachman (1987), Department of Defence Architecture Framework (DoDAF) (Department of Defence 2010), Federal Enterprise Architecture Framework (FEAF) (CIO Council 2001), and The Open Group Architecture Framework TOGAF 9.1 (The Open Group 2013). In this section a brief overview is provided of these four frameworks. Architecture framework shows the different architectures, people, processes, tools, and products in the enterprise. Framework provides guidelines on how these elements in the enterprise work together. Therefore, this overview helps in understanding what EA is and how it works.

1. John Zachman defined Zachman enterprise framework as "a schema - the intersection between two historical classifications that have been in use for literally thousands of years. The first is the fundamentals of communication found in the primitive interrogatives: What, How, When, Who, Where, and Why. It is the integration of answers to these questions that enables the comprehensive, composite description of complex ideas. The second is derived from reification, the transformation of an abstract idea into an instantiation that was initially postulated

by ancient Greek philosophers and is labelled in the Zachman Framework: Identification, Definition, Representation, Specification, Configuration and Instantiation" (Zachman 2008). Zachman framework EA perspectives are organised in two dimensional matrices such that the rows represent the type of stakeholders, and the aspects of architecture are represented on the columns. This framework doesn't represent a methodology that can be used by organisations; rather, organisations need to fill up all its fields of their own interests (i.e. processes, rules, events, and so on). There are no priorities of the columns. However, the top-down order of its rows aligns with business enterprise. Each cell has its own details; for example for IT people, the details would be applied equally to the physical materials (i.e. transformers, piping, and so on) and their associated roles, processes, locations, and so on. So, information system architecture does not define the details of the models (Wegmann et al. 2008).

2. Department of Defence Architecture Framework (DoDAF) was created by the United States department of defence (Department of Defense 2010). It provides different views depending on stakeholder's concerns. The different views can be: visualisation of different artefacts, complexities and scopes of the projects such as structural, ontological, tabular, temporal, and so on (Working Group 2007). DoDAF suits the large complex systems with interoperability challenges between different perspectives of the enterprise. It offers operational views with details for specific domains and the interactions with other domains (Working Group 2007). Although DoDAF is aimed at military systems, it has been applied to many fields such as public, private and voluntary areas in different countries around the world (Department of Defense 2010). DoDAF uses a meta-model framework that defines the attributes in each view along with their interaction relationships in the model. In general, it includes three layers: (1) the conceptual data model (CDM) which defines the high level of data constructs that represent the base for the architecture, which help the executives, seniors and managers in understanding the basis of the data, (2) the logical data model (LDM) which adds and defines the different attributes as well as their interrelationships in the model, and (3) the physical exchange specification (PES) that consists from LDM as well as implementation attributes such as data source and date, and the general data types, which are all added together. However, DoDAF doesn't offer the physical data

model. This means that the software developer will be responsible for implementing the principles and practices for DoDAF (Wegmann et al. 2008).

3. The federal CIO council published the Federal Enterprise Architecture Framework (FEAF) in 2001 (CIO Council 2001). The FEAF provides guidance and standards for developing and documenting architecture descriptions for multi-organisational functional segments of the federal government. Four models were introduced to FEAF: (1) business reference model (a function-driven framework for describing an organised, hierarchical construct day-to-day business operations), (2) service component reference model (a business and performance-driven functional framework that classifies service components with respect to how they support business and/or performance objectives such as customer services, business management services and support services), (3) data reference model (an aggregate description of data, information and the interaction between different stakeholders), and (4) technical reference model (a component-driven and technical framework that categorises the technologies and standards that support and enable service delivery).

4. The Open Group Architecture Framework (TOGAF). TOGAF is one of the most comprehensive EA that includes planning, designing, implementing, and governing the enterprise phases (The Open Group 2013). Open sources approach has been used to build this framework such that a large number of architecture practitioners have been included. Figure 4.1 shows TOGAF 9.1 (The Open Group 2013) architecture development method (ADM), which forms the heart of TOGAF. This diagram explains the method of developing and managing the lifecycle of EA (The Open Group 2013). It integrates most elements of available architectural assets and the needs for businesses and IT organisations.

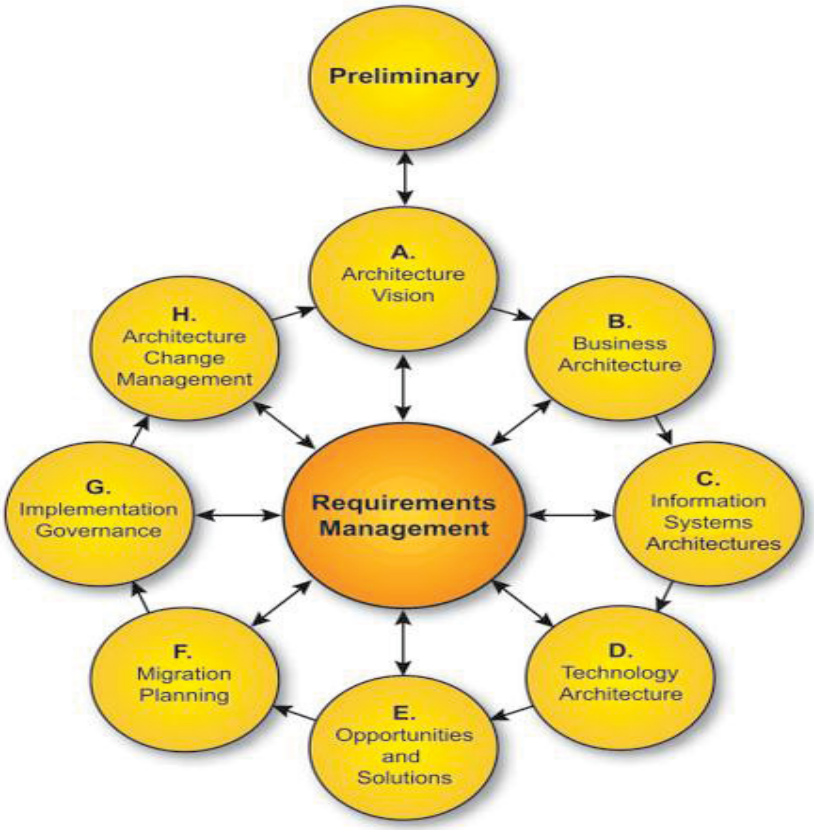


Figure 4.1. TOFAF 9.1 ADM (The Open Group 2013)

ADM is an iterative and cyclic process that was designed to deal with organisational and systematic requirements. ADM can be extended to meet individual and specific needs (The Open Group 2013). The phases of the ADM are divided into sub-steps for example phase D (technology architecture) is divided into: identifications, tools, models and viewpoints (The Open Group 2013).

TOGAF 9.1 has four architecture domains (The Open Group 2013): (1) business architecture, which defines the organisation's strategy, governance and key process, (2) application architecture, which defines the individual application systems, how to be developed, their interactions and relationships, (3) data architecture, which describes the physical and logical data assets in the organisation and the management resources for that data, and (4) technical architecture, which describes the tools, technologies, software and hardware, and the needed infrastructure.

However, TOGAF 9.1 doesn't define a clear time-frame for iterations, refers to commitment as explicit to all stakeholders without telling how to gain this commitment, and introduces continuous requirement without providing the continuous adaptation activity for EA model (Buckl et al. 2011). Also, before applying ADM, its components need to be modified or extended to cover specific needs such as emerging the social domain (Gill 2013b).

4.3.2 Agile Enterprise Architecture

EA and agile methodologies have some conflicting goals. In part, the goals come "about because of both differing focal points (process repeatability and documentation for EA vs. adaptive change and short delivery cycles for Agile) different starting points in the development process (learning for Agile, documentation for Enterprise Architecture), as well as different process flows" (Schulmeisters 2014). However, their fundamental goals are similar such that they both seek to create solutions for stakeholders, customers, business leadership, and technology managers alike. Also, both share the way that maximises stakeholder value, and manage change and complexity in a way that minimises constraints on future efforts (Schulmeisters 2014). If EA and agile methodologies are balanced against each other and placed in the right context with respect to each other, they can enhance the achievement of those goals. EA as a framework and customer of agile brings a rigor, consistency and overarching vision that reduces the likelihood of an agile process failing to test and document the changes it makes along the way, and agile methodology applied to implementation of EA increases the perception of value delivered by EA, and speeds up the achievement of tangible results (Schulmeisters 2014).

Enterprise is said to be agile when it is "responsive (scans, senses and reacts appropriately to expected and unexpected changes), flexible (adapts to expected or unexpected change at any time), speedy (accommodates expected or unexpected changes rapidly), lean (focuses on reducing waste and cost without compromising on quality), and learning (focuses on enterprise fitness, improvement and innovation)" (Gill 2013a, p. 3).

Agile architecture should have the ability to respond to changes effectively and efficiently, rather than using available architectures to fine-tune understanding of the system(s) to handle potential changes (Nair 2008). Agile architecture should have the ability of extensibility (add new functionality), modularity (substitute one modules with another without disrupting the integrity of the rest); and reconstitutability (recombine, or substitute modules in the face of a fault or failure) (Madni 2008).

Agility cannot be achieved by only focusing on improving the operations of organisation (Gill 2015b). Unlike traditional process and documentation focused EA, agile EA offers an incremental and people (users and customers) focused approach which is aimed at enhancing agility (Gill 2015b; Ross et al. 2006; Venkatesh et al. 2007). That is, people have a big role to play in an organisation and agility is not only the outcome of technological achievement, advanced organisational and managerial structure and practice, but also an outcome of human abilities, skills, and motivations (Mthupha 2012). Therefore, agile EA should respond to changes effectively and efficiently (Batra et al. 2010; Gill 2013a; Tamm et al. 2011) and should be easily, rapidly, and continuously communicated (Ambler 2014b; Edwards 2006). In addition, the scope of agile EA should include people, processes, information and technology of the enterprise, and their relationships to one another and to the external world (Gill 2014a).

Failure in architecting a project leads to failure to deliver this project. In addition, designing new architecture, from scrap, for every project is a failure to deliver projects (Ambler 2014b). The difference between the two failures and the middle way of success is agile architecture (Ambler 2014b). One goal of agile EA is to make sure that all systems are built so it can fit into the existing and future development environment (Leffingwell 2007).

Accordingly, agile EA can, simply, refer to doing EA in the agile way with special focus on stakeholders involved (Ambler 2014b). Agile EA can also be defined as “the systematic process of adhering to agile development principles while translating business vision and strategy into effective enterprise change by flexibly creating, communicating and improving key requirements, principles and models” (Mthupha

2012, p. 49). Models should have agile characteristics embedded in them, describe the enterprise's future state, and keep options open as late as possible and enable its evolution (Mthupha 2012).

In this study, we define agile EA as "a blueprint that describes the overall structural, behavioural, social, technological, and facility elements of an enterprise's operating environment that share common goals and principles with the ability of responsiveness, flexibility, speediness, leanness, and learning" (Gill 2013a).

Buckl et al. (2011) summarise four common challenges for EA; stakeholder's satisfaction, customer's requirements, stakeholders' commitment and the flexibility to requirement changes. They argue that none of the traditional EA (i.e. Zachman, DoDAF, FEAF, and TOGAF) can deal with these challenges in ASD enterprise. According to Ambler (2014b), agile EA has two organisational structures: (1) formal structure: this form is documented by organisation such as chart, and (2) informal structure: this form is not documented, but rather used by developers to get the things done or what is called "go to guys", which means looking for other developers who have critical skills or knowledge, which is what agile it trying to be. A common problem or thread of using traditional EA is a focus on tools and processes over stakeholders' interactions (Ambler 2014b). To overcome the above challenges, agile EA has to meet the following features (Ambler 2014b; Buckl et al. 2011):

1. focus on people, not technologies or techniques,
2. keep agile EA design as simple as possible,
3. the incremental nature of agile methods allows developing different EA products and the iterative nature allows reacting quickly to changes. Agile EA needs to be incremental,
4. as a result of daily interaction between developer and customer, any problem raised has to be addressed and dealt with instantaneously in agile EA,
5. work with developers in field to gain better understanding of EA needs. Developers can help by addressing requirements head on, which gives

development teams an opportunity to address components that might not be working in the best way or aren't complying with architectural principles,

6. look at the whole agile EA picture,
7. make agile EA attractive to stakeholders, and
8. agile EA promotes the idea of self-directed and self-organised members, which helps to keep administrative overhead to the lowest level as the agile EA should be.

An agile EA should have capability, which is "the ability of an entity or enterprise that applies different methods, practices, models and tools to support enterprise adaptation" (Gill 2012). ASD requires multiple enterprise capabilities and stakeholder groups to be engaged. These groups usually have different requirements, objectives and perspectives. Agile EA can be used to enhance the modern complex enterprise adaptation (Gill 2013b). Most of the enterprises focus on adopting a solution for the weakest attribute in isolation from the other attributes and in local environment, which might reflect negatively on the other attributes. This approach (i.e. solution for weakest attribute), if applied on GDAD, will have a greater negative impact on the enterprise (Gill 2013a). Hence, looking at the enterprise as a whole would be a less risky and more effective approach (Gill 2013a). Moreover, Gill (2013b) reported that enterprise needs to be viewed as an agile or adaptive system in order to deal with the complexity and the instability of current environment. In other words, agile EA needs to be re-defined and adaptive to adaptation initiatives such as establishment and integration (Gill 2013b).

4.3.3 An Overview of Agile EA Frameworks

Software development organisations face the challenge of surviving in such an unstable business environment, changeable customer requirements, unstable market structures, and changeable laws and policies (Herbsleb 2007). In fact, software organisations are open systems that evolve through interaction with the surrounding environment (Gill 2015b). Agility and adaptability is required for these organisations to survive in such a complex environment.

To address this challenge, various organisations have offered different suggestions and approaches to change from traditional EA into agile EA. For example, MIT Sloan School of Management's (MIT 2006) offers "Business Agile Enterprise", the Agile Alliance (Vickoff 2014) offers the PUMA methodology, and Gartner suggests blending Lean and Agile methodologies (Wilson 2013).

Some agile or adaptive EA frameworks have been introduced such as the adaptive EA framework (a.k.a. The Gill Framework), Hewlett Packard (HP) Adaptive Enterprise Architecture (Di Vitantonio et al. 2006), and the KASRA Framework (Najafi & Baraani 2010). Here, HP and KARSA will be briefly defined, and The Gill Framework which represents the most comprehensive framework, will be discussed in the following section. Table 4.1 summarises the differences between traditional EA frameworks and agile EA frameworks. The above four traditional frameworks (i.e. Zachman, DoDAF, FEAF, and TOGAF, discussed in Section 4.3.1) lack adaptability and agility. In addition, these frameworks lack the academic background and lack the support of guidelines to develop enterprise context and the predefined models of the enterprise context before the actual commencement of EA (Gill 2013a). Moreover, the enterprise architecture practices defined by these frameworks are "often considered to be overly bureaucratic, time consuming, expensive and document driven" (Gill 2015b, p. xvii).

HP framework applies a data-centric approach where hardware, software, and services are brought together in order to efficiently and effectively respond to change (Di Vitantonio et al. 2006). IT resources are optimised to achieve the business performance indicators and operations. This is achieved by synchronising the business processes, the information and communication technology services, infrastructure, and applications, which lead to more effective response to change (Akhigbe 2014).

KASRA Framework is based on the service-oriented architecture framework (Najafi & Baraani 2010). The framework involves a service-oriented lifecycle that is compatible with information technology infrastructure library best practices as well as a classification schema (Najafi & Baraani 2010). The classification schema encompasses a collection of different viewpoints (rows) and aspects (columns) of the enterprise, which constitute a comprehensive overview. The schema describes different aspects of

an organisation from different viewpoints (i.e. strategist, business leader, business user, and IT specialist) against different aspects of the service lifecycle (purpose, policy, pattern or practice, stakeholder, people, and resource). The meta-model provides a holistic view of relationships between KASRA framework entities during lifecycle, which represents the dynamic model of the framework (Akhigbe 2014).

Table 4.1: Comparison between agile EA and traditional EA frameworks

	Agile EA	Traditional EA
Focus	• Enables enterprise service system agility by building processes and applications over an IT infrastructure that is agile and leveraging services as key elements in addressing agility • Supports the IT strategy, business and IT alignment, and IT support for business innovation and high performance • Guides how enterprise requirements, strategy, architecture, project and enterprise service management disciplines fit together and can be used to meet requirements with high performance outputs	• Focuses on where an organisation is going and how it will get there • Concentrates on bringing together business owners, information specialists and technology implementers • Connects the organisation units as a whole: business processes and capabilities, IT systems, and their interrelationships • Manages the requirements (quality) and main them over the period of its useful life
Approach	• Uses of sets of integrated adaptive disciplines of overall structural, behavioural, social, technological, and facility elements of an enterprise's operating environment that share common goals and principles for developing and managing EA adaptation • Applies a flexible and human-oriented approach to respond to change by integrating EA with other related enterprise disciplines such as enterprise strategy, projects, services and requirements management which leads to faster response to business needs	• Guides the evolution of an organisation's systems and processes towards achieving desired levels of consistency, effectiveness and efficiency • Uses to depict the current states of the organisation and its desired future state • Informs, guides, directs, and constrains the decisions taken by human beings within organisations • Provides a long-term view of an organisation's processes, technologies, and systems, which enables individual projects to build capabilities
Domains	Interaction architecture, facility architecture, solution architecture, business architecture, information architecture, application architecture, platform architecture, infrastructure architecture, social architecture	Business architecture, application architecture, information architecture, technical architecture, product architecture, data architecture
Elements	• Defining, operating, managing, supporting, context, rationalisation, assessment, realisation, and unrealisation capabilities • Strategic objectives and operational	• Structure, functions, behaviour of its people, processes, technologies and concerns such as security, accessibility, privacy and interoperability

	performance measures	
Adaptability	• Agile EA is responsive, flexible, speedy, lean, and focuses on enterprise fitness, improvement and innovation • Integrates enterprise strategy and architecture to facilitate strategic assessment and adoption or de-adoption of technology in response to constantly changing business • Simplifies and speedup the complex non-technology and technology-driven enterprise adaptations	• Not adaptive or it can be adaptive in local domains separately but not in the organisation as a whole, as the business evolves. This leads to a lack of integration between business objectives, processes and resources • Provides little room for flexibility • Any change in any of the processes costs a lot in terms of resources such as time, workforce, cost, and causes diverse effects on the business
Social architecture	Provides social architecture which involves social structure, culture, behaviour, knowledge and opinions of community in an organisation	Does not provide social architecture

4.3.4 Adaptive EA Framework

Adaptive EA or The Gill Framework, as an agile meta-framework, has been developed in response to traditional EA frameworks drawbacks and is a mix of both theory and practice (Gill 2012). It extends and complements TOGAF 9.1. This framework has filled the gaps in TOGAF 9.1 (i.e. provides the social architecture domain, commitment and adaptation activity) and provided the academic basis along with industrial standards (Gill 2013b). This framework along with existed tools, provide a practical support to agile EA. As shown in Figure 4.2, this framework has two main layers (Gill 2012):

1. The outer layer: it helps in assessing or adopting emerging technologies such as cloud and social technologies. It represents guidance for continuous adaptation of agile EA in response to external and internal changes. It defines five EA adaptation capabilities (i.e. context awareness, assessment, rationalisation, realisation and unrealisation). (1) Context awareness capability defines the tool or technology to be used in EA such as the "Force.com cloud-based chatter communication tool", which was used to assess the GDAD communication in (Gill & Bunker 2013). (2) Assessment capability defines the technology that needs to be assessed before used in agile EA such as the communication technology assessment tool (CTAT), which was assessed using The Gill Framework (Gill et al. 2012). (3) Rationalisation capability describes the identification, selection and modelling of the services that are going to be used for agile EA along with their adoption motivators and de-

motivators such as XaaS (e.g., service requirements analysis: software as a service, platform as a service) (Gill et al. 2012). (4) Realisation and Unrealisation capabilities define the acceptance or rejection for the specific technology or service that is going to be used in agile EA (Gill 2012).

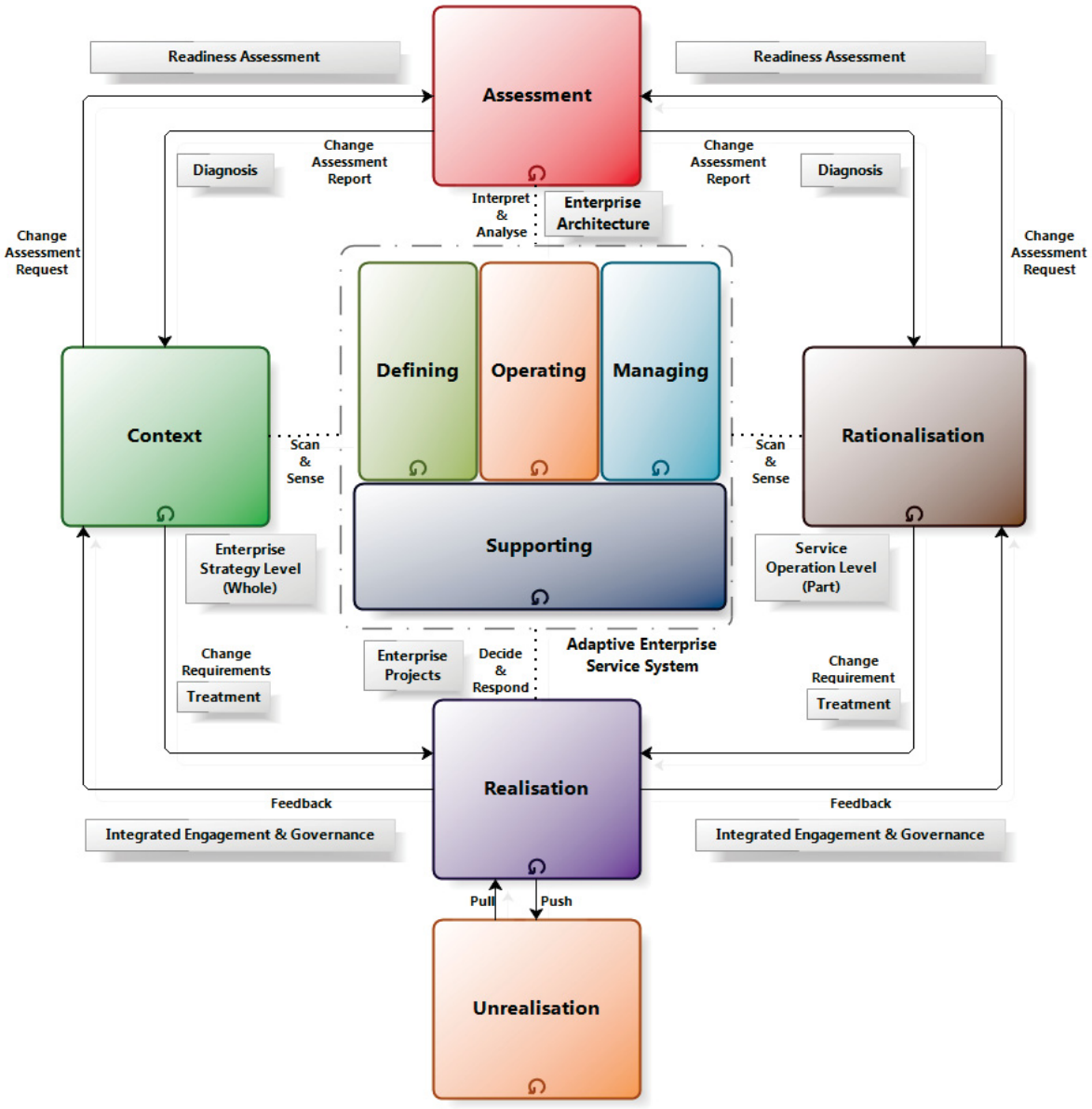


Figure 4.2. The Gill Framework (Gill 2014b)

2. Inner layer: It assists the four capabilities (i.e. defining, operating, managing and supporting). (1) Defining capability defines agile EA capability such as Adaptive Enterprise Service System (AESS) model, which was presented in the work of Gill (2013a). (2) Operating capability describes agile EA when it operates (process)

regarding interaction, factory and facility architectures. (3) Managing capability is a set of integrated capabilities such as procedures, strategies, projects, services and requirements management that are used to develop and manage the agile enterprise. (4) Supporting capability describes integrated capabilities such as enterprise operating model, intelligence, asset library, supply chain, and method engineering. These capabilities support the other capabilities in the agile enterprise.

As it was introduced above, The Gill Framework covered the "social architecture", which refers to social structure, culture, behaviour, knowledge and opinions of community in an organisation (Gill 2013b). Social architecture is important for agile EA as it explains and guides the “social communities”, which includes all ongoing social operations, goals, objectives and activities. Communication is a part of social architecture or, in fact, it is the core of the social architecture. Additional elements can be added to the social architecture if required, which give more flexibility to The Gill Framework. In fact, The Gill Framework, in a general sense, has the ability to be extended or expanded whenever there is a need for that (Gill 2012), which gives it another competitive feature above the other frameworks.

The agile EA domains, in The Gill Framework®, include (Gill 2012): interaction architecture (describes the external interaction of different systems), factory architecture, facility architecture and solution architecture. Solution architecture describes how all the separate EA viewpoints come together or are integrated into implemented systems or solutions (Gartner 2008). Facility architecture is defined as the “fundamental concepts or properties of a facility system in its environment embodied in its elements, relationships, and in the principles of its design and evolution” (Gill 2012). Factory architecture includes human architecture (e.g., guides relationships between different people in the enterprise) and IT architecture. IT architecture includes application architecture (e.g., guides application design, data structures, and applications structure), platform architecture (e.g., storing, delivering, and optimising a variety of information) and infrastructure architecture (e.g., structure and behaviour of the technology infrastructure). Human architecture includes social architecture, business architecture (e.g., business goals, business functions or capabilities, business processes, and roles) and information architecture (e.g., organising and labelling different

information). Social architecture refers to social structure, culture, behaviour, knowledge and opinions of the community in an organisation (Gill 2012).

4.4 Agile EA Driven Communication Approach

Figure 4.3 illustrates GDAD sub-team's arrangement. In GDAD sub-team, organisation may have a team of project management team, Product Owner team, and architecture owner team. Depending on the size of the organisation, each team may have team lead, Product Owners, architecture owner, and team of developers. Product Owner may represent the customers in big organisations, or customers may be included in the team.

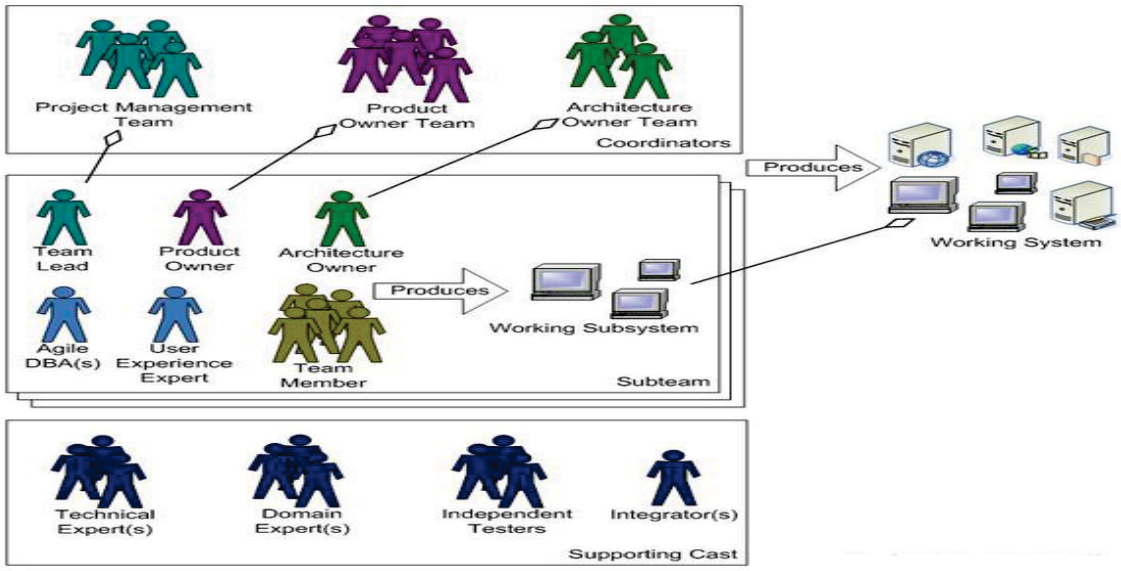


Figure 4.3. Geographically distributed agile team's structure (Ambler 2014b)

Agile principles explain that the best architectures and designs emerge from self-organising teams, and emphasise that business people and agile developers must work together daily throughout the project (Agile Manifesto 2001). These two principles work well for a small co-located agile team where team members and business people work out the best project architecture and design through active communication and continuous collaboration (Ambler 2014b). However, in GDAD environment, these two principles would be hard to achieve (Batra et al. 2010; Ramesh et al. 2006). In such complex GDAD environment, different silo GDAD teams need to be continuously, efficiently and effectively communicated with different changing to their and other dependent project(s) architectures and requirements for alignment (Ambler 2014b; Gill

2015b). This could be achieved by using the overall agile EA holistic integrated shared view (Boh & Yellin 2006; Tamm et al. 2011) along with using the available communication tools. Agile EA as a whole evolves which means that up-to-date real-time information about integrated business and software architecture elements are shared between GDAD teams. This provides a holistic shared view to GDAD teams as the different parts of it (e.g. business and IT capabilities, services and products) are developed and delivered by the GDAD teams in different increments at different times. Therefore, as discussed above (Section 4.2), this shared view will increase the common ground (i.e. agile EA can be seen as an enabler and a tool of common ground that decreases the effects of the communication constraints or challenges) among distributed GDAD teams (Niemi & Pekkola 2015). This approach is discussed in the following paragraphs.

Agile EA aims at producing enough architecture to meet the project's needs (not more) and finishing the project as scheduled with minimum effort. Agile EA should be a team effort following the strategy of "everyone owns the architecture". However; this strategy has some problems such as (1) people do not agree on the architecture sometimes, which make it necessary to have someone to be in the role of architecture owner to facilitate agreement, and (2) GDAD sub-teams need to be organised, which requires someone to be in a coordinating role. Just like Scrum's Product Owner who assigns team's requirements, the architecture owner, as a member of each GDAD sub-team, is responsible for team's architecture, acts as a liaison to the EA team, and act as a consultant to the sub-team's members (Batra et al. 2011).

As each agile method has its own practices and artefacts, this study explains five practices of agile development lifecycle building on XP and Scrum methods. These practices are: planning (adaptive), requirements practice, decomposition, Sprint (small release), and working product (tested bug free product) (Madison 2010).

In Ch.3, six categories and seventeen challenges were identified and discussed for GDAD communication. The six categories and their challenges are:

C1: Distance Differences (Time-zone differences, Geographic differences)

C2: Team Differences (Team size, Number of teams, Coordination)

C3: Customer Communication (Customer involvement, Customer representative involvement)

C4: Project Characteristics (Project domain, Project architecture)

C5: Organisational Factors (Project management process, Communication tools, Communication infrastructure, Organisational culture

C6: Human Factors (Language, National culture, Trust towards teams or team's members, Personal practice)

Agile EA should be communicated through the different enterprise project levels of Adaptive Enterprise Project Management (Gill 2014c), namely; iteration, release, project, and program level. Figure 4.3 explains how agile EA can be used in GDAD environment. We have different architectural views according to the different architectural (enterprise project management levels) levels. (1) Distributed teams (up to N teams) share the "project architecture view". (2) Different projects (up to N projects) share the "program architecture view". The same is applied to the "solution architecture view", which can have "N" number of program architectures. (3) Each of architectures updates the Architecture above. All architectures are then updated and shared using the enterprise architecture integrated knowledge base. This knowledge base can be represented in multiple repositories which grant access to all GDAD stakeholders. This way ensures that all stakeholders are updated with the latest changes (i.e. project or program changes). Communication challenges affect all communication's levels (i.e., programs, projects and teams' communication).

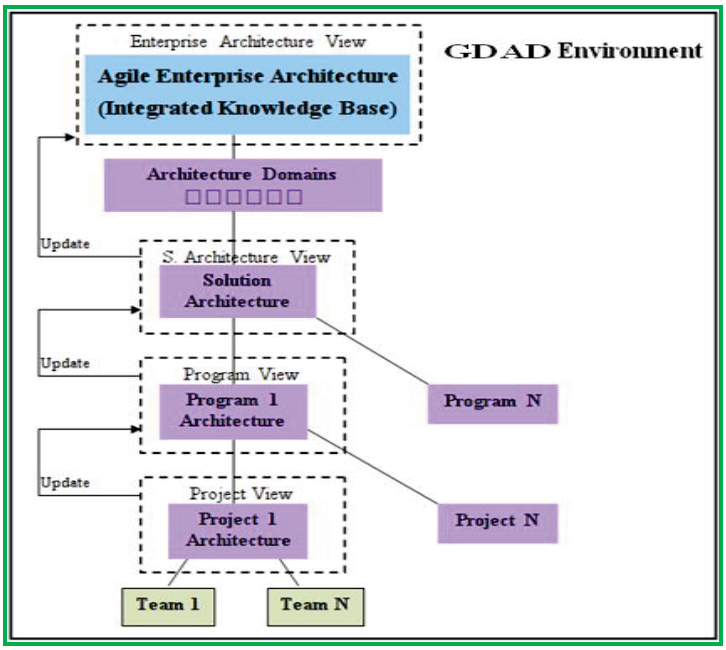


Figure 4.4. Agile EA driven GDAD approach

Having in mind the above five agile practices, the above six GDAD communication challenges, Figure 4.4, and the following scenario, Section 4.4.1 and Section 4.4.2 discuss how agile EA can enhance GDAD communication and mitigate the effect of GDAD communication challenges:

- First scenario-single GDAD project: the architecture owner shares solution architecture among different teams or team's members on the same project. He/she also updates the agile EA, regularly, after completing the project or part of it. All teams and team's members share the team or individual artefacts through using the common view of agile EA (i.e., knowledge base).

- Second scenario-multiple GDAD projects: different DAD projects (program level) have different architectures owners; one for each project. They share different projects' artefacts through using the common vision of the agile EA knowledge base. All projects' artefacts are shared and can be accessed using agile EA knowledge base.

4.4.1 Architecture Owner Roles[2]

The role of architect (i.e., architecture owner) is important especially in the case of big teams. Architecture owner knows better than the other team members about the organisational structure, solution architecture and the trade-off that may take place for the best of the business (Madison 2010). Architecture owner represents the agile EA in GDAD project. He/she plays an important role in all GDAD practices. Architecture owner roles start even before the other developer's roles in the "planning" practice or what it called adaptive (all developers share). However, the meaning of planning or up-front practice is not the same as for traditional SD. Here, architecture owner needs to have an idea about the organisation EA before starting a project. For example, for such a project, the organisation may have two or three designs to develop this project. The role of architecture owner in this case is to tell the agile team about these options and work them out together to choose the best option. This action gives the team high agility within the organisation EA. Table 4.2 summarises architecture owner roles among the above five practices. Moreover, the role of architecture owner is discussed among the role of agile EA in each agile practice.

1. Planning (adaptive) practice: the main role of agile EA is to provide the GDAD developers with the holistic architectural direction setting (Madison 2010). Agile development is done in small iterations and encourages all members to collaborate to find better solutions later during development; however, the pre-options of solutions and design patterns increase the agility, save time, and give an overall direction (Ambler 2014b) [**C5**]. Agile developers, liaison by architecture owner, are offered different options rather than specific solutions, design patterns, high-level diagrams, components reuse, quality and trade-off attributes [**C4**], and initialisation to communication channels [**C3, C5**]. Agile EA is concerned with facilitating independent development by team's members and minimising the number of unproductive hours [**C2**]. Architecture owner keeps GDAD teams updated with the interface changes, what backward compatibility is required, what functionality to build and the other component interfaces to develop against. As a principle, no team can initiate development on functionality that depends on the functionality being

[2] Sections 4.4.1 and 4.4.2 appeared in Alzoubi et al. 2015

developed by another team [**C2**]. Although this principle slows down the development process, the removal of coordination cost and the short cycles for agile team outweighs any benefit that may be achieved by current development.

2. Requirements practice: the main role of agile EA is to help in getting agile team members on board and structuring business and architecture needs (Madison 2010). Customer requirements that have significant foundational or directional influence can be identified [**C3**]. Architecture owner can help team members to correct deviation or fine tune architecture [**C4**]. Agile EA helps Product Owner to prioritise the customer's requirements with the business user requirements and build them in conjunction with the business functionality in each Sprint [**C3**]. Architecture owner works with Product Owner to estimate the effort that an item in the Product Backlog will take, which means assigning the item or sub-item to the relevant developer [**C1**].

3. Decomposition practice: agile EA helps Product Owner to identify boundaries of the architecture, and determine business value of each item (part of the product) [**C4**]. Architecture owner may help in decomposing the architecture into development tasks that can be assigned to relevant developers.

4. Sprint (small release) practice: agile EA helps to ensure that the Sprint functionality has been met and keep the Sprint on the track (Madison 2010). Agile team commits itself to implement a specific Sprint goal selected during the daily meet [**C6**]. Daily meets discuss: what has been done, what is the current status, what is next to be done and if there is any problem. This helps in receiving early feedback to deal with any change or intervention needed and maintain early decisions and clear focus of agile team (Buckl et al. 2011) [**C4, C5**]. Architecture owner collaborates with the team during the Sprint helping in design issues and business objectives, and advising and offering consultation to team's members [**C6**].

5. Working product (tested bug free product) practice: agile EA helps in measuring the architecture state of the product (Madison 2010). Developers measure architectural attributes; code, functionality, and refactoring, integration and testing for each

Sprint release in order to make sure that the customer's needs were met [**C3**]. Architecture owner may start reviewing the working product several days before the official review to ensure its functionality, then, work on code and architecture documentation.

Table 4.2: Architecture owner roles among agile project

Practice	Role
Planning (adaptive) (Setting architectural direction)	• Establish design patterns • Generate high-level diagrams • Establish components reuse • Apply quality and trade-off attributes • Initialise communication channels • Offer different solutions
Requirements Practice (Getting members on board and structuring business and architecture needs)	• Apply direction influence • Correct deviation or fine-tune architecture • Apply business function
Decomposition Practice (Task assignment)	• Work with Product Owner to identify boundaries of architecture significance • Work with Product Owner applying business value and refactoring
Sprint Practice (small release) (Building functionality)	• Help in design issues and business objectives • Advise and offer consultation
Working Product Practice (tested bug free product) (Measuring architecture state)	• Work on code and architecture documentation • Measure architectural attributes; code, functionality, refactoring

4.4.2 Agile EA Driven Communication Roles

In fact, GDAD projects have two complementary communication needs; the formal communications for crucial tasks (e.g., updating project status and escalating project issues), and informal communication in order to keep team members aware of the information that would enable them to work together efficiently (Lanubile 2009). The approach presented here is to move any coordination needs from the team level to the architecture (Bosch & Bosch-Sijtsema 2010). GDAD should have a communication system that could continuously feed GDAD teams with holistic architectural and strategic information to keep them on track (Gill 2014d). GDAD teams need to be decoupled as much as possible regarding design rules and organisational structure. The spread of design features across sites would increase the interactions among teams for

agreement on design requirements. This suggests that there could be a conflict between teams about solution design. This can be addressed by using architectural description (Martini et al. 2013) to keep all teams on the same page. This description would allow GDAD teams to focus on customer's needs, avoid organisational dependencies caused by architectural dependencies, and satisfy customer's needs but keep in mind the organisational software design and include reusability as an important quality. Table 4.3 summarises the impact of agile EA on GDAD project and communication among the five agile practices discussed above.

Table 4.3: Impact of GDAD communication using agile EA on GDAD project (adapted from Madison 2010)

Practice	Impact
Planning	• Understand business objectives • Get input from agile team • Understand the business vision and directions
Collecting	• Facilitate story session • Use user's stories to build design patterns, improve refactoring, and validate hardware and software
Decomposition	• Maintain architecture significance • Maintain business value
Sprint	• Build functionality • Support agile team
Working product	• Review documents • Advocate refactoring

4.5 Research Model Constructs[3]

This section discusses the relevant literature and identifies three constructs of the research model: agile EA (including one antecedent or independent variable: agile EA), GDAD active communication (including two dimensions or dependent variables: efficiency and effectiveness), and GDAD performance (including four dimensions or dependent variables: on-time completion, on-budget completion, software functionality and software quality). The central construct of the research model is GDAD team communication. The research model and related hypotheses are shown in Figure 4.4.

[3] This part of work appeared in Alzoubi & Gill 2015; Alzoubi & Gill 2016; Alzoubi & Gill 2017a, b

This model was rigorously build based on literature review (Ch.2, Ch.3, Ch.4) and preliminary evaluation (Ch.5). The adopted working definitions for all the research constructs are summarised in Table 4.4.

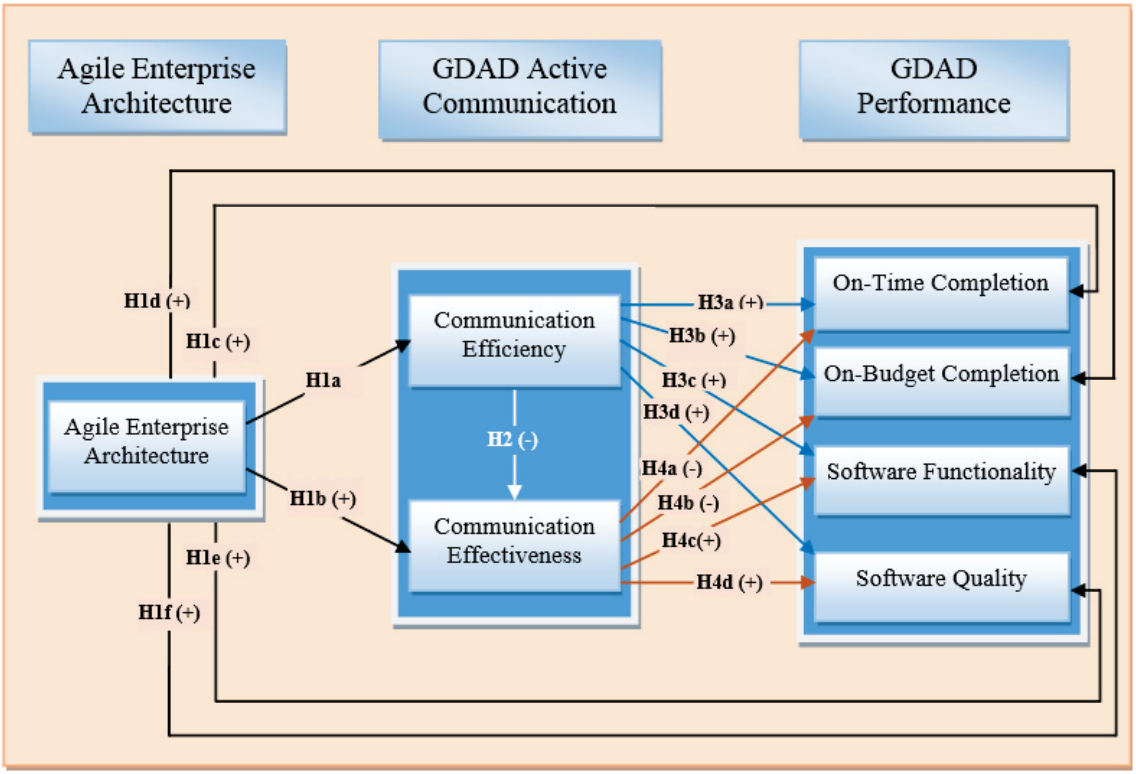


Figure 4.5. Research model

Table 4.4: Summary of the theorised constructs and their definitions

Construct	Definition/ Description	Literature
Agile EA	A blueprint that describes the overall structural, behavioural, social, technological, and facility elements of an enterprise's operating environment that share common goals and principles with the ability of responsiveness, flexibility, speediness, leanness, and learning	Gill 2013a
Communication efficiency	Delivering a message to a receiver with high quality and with minimal time, cost, effort, and resources required to establish communication	Alzoubi & Gill 2015
Communication effectiveness	Delivering a message to the receiver who understands it as it was intended with minimal disruption and misunderstanding, even if it takes a long time	Alzoubi & Gill 2015
On-Time completion	The extent to which a software project meets its baseline goals for duration	Aladwani 2002; Lee & Xia 2010;
On-Budget completion	The extent to which a software project meets its baseline goals for cost	Aladwani 2002; Lee & Xia 2010
Software functionality	The extent to which the delivered software project meets its functional scope goals, user needs, and technical requirements	Lee & Xia 2010
Software quality	Delivering a good working product that satisfy the requirements stipulated in the problem statement and produce correct results	Balijepally et al. 2009; Chow & Cao 2008

4.5.1 Agile EA Construct

EA has been used for many years by traditional SD organisations (Avritzer et al. 2010). According to Bass and Kazman (1999), using EA development "differs from traditional development in that it concentrates on driving design and maintenance from the perspective of software architecture. The motivation for this change of focus is that software architecture is the placeholder for system qualities such as performance, modifiability, security, and reliability. The architecture not only allows designers to maintain intellectual control over a large, complex system but also affects the development process itself, suggesting (even dictating) the assignment of work to teams, integration plans, testing plans, configuration management, and documentation. In short, the architecture is a blueprint for all activities in the software development life-cycle".

The effective use of EA standards can provide cost and efficiency advantages by standardising the different platforms, technologies, and application architectures among

distributed sites. This can potentially reduce the organisational operational complexity, minimise waste and replication of system components, enable reuse of system components, and control the number of skilled individuals (e.g., developers) required to maintain the systems (Boh & Yellin 2006). Moreover, using EA standards enables integrating applications and sharing data across distributed sites. This will enable distributed sites to integrate their business processes, develop key applications more rapidly, and make effective use of organisational data (Boh & Yellin 2006).

Although empirical studies show a strong relationship between social and technical architectures in software enterprise (e.g., Cataldo et al. 2008), they only offer a narrow assessment of the coordination mismatches inside an enterprise. The management skills and means of how tasks inside an enterprise can be partitioned and assigned to maximise the outcomes depending on the inter-relations between different enterprises are often ignored or uncovered (Avritzer et al. 2010). There is a lack of theoretical EA frameworks that can predict communication and coordination needs, which can assist management to prevent mismatch occurrence (Ali Babar et al. 2009; Avritzer et al. 2010; Gill 2013b).

Ambler (2014b) summarises benefits of using agile EA as follows:

1. avoiding chaos that will result if team members think that they can do whatever they want using any technology they want,
2. avoiding duplication of functionality and information,
3. achieving continuous and effective design reuse,
4. enhancing better integration between different systems,
5. avoiding conflict between systems, which might cause system failure, and
6. decreasing development price.

There are also some disadvantages to agile EA that must not be overlooked. These disadvantages are (Ambler 2014b; Leffingwell 2007; Mthupha 2012):

1. agile EA does not include an explicit way to ensure compliancy (although having enterprise architects embedded in the teams goes a long way towards this),
2. it relies on people being responsible,
3. it requires the team to do their best to keep things simple, and
4. it requires and assumes team to model and document the EA.

Table 4.5 summarises the two streams of literature body about using agile EA in agile methods: principles of agile methods stream and research stream (empirical and experience research). ASD has promised to work in high-level uncertainty environment with changeable requirements (Agile Manifesto 2001). Agile EA should respond to changes effectively and efficiently rather than using available architectures to fine-tune understanding of the system(s) to handle potential changes (Batra 2009; Gill 2013a; Tamm et al. 2011). Therefore, agile EA should be easily, rapidly, and continuously communicated (Ambler 2014b; Edwards 2006). Agile principles explain that the best architectures and designs emerge from self-organising teams, and emphasise that business people and agile developers must work together daily throughout the project (Agile Manifesto 2001). These two principles work well for a small co-located agile team where team members and business people work out the best project architecture and design through active communication and continuous collaboration (Ambler 2014b).

The importance of agile EA would be critical in GDAD environment due to many challenges (as discussed in Ch.3). In GDAD, a viable and accepted EA strategy is critical (Ambler 2014b). To keep that efficiency and to deal with current quickness and complexity, GDAD enterprises need to be constantly adopting a contemporary agile approach (Gill 2013a). The above principles of agile development would be hard to achieve in GDAD (Batra et al. 2010; Boehm & Turner 2003; Ramesh et al. 2006). GDAD teams need to be continuously, efficiently and effectively communicated with different changing to their and other dependent project(s) architectures and requirements for alignment (Ambler 2014b; Gill 2015b). This could be achieved by using the overall agile EA holistic integrated shared view (Highsmith 2000) along with using the available communication tools.

Table 4.5: Literature review and agile methods principles about agile EA

	Literature	Relevant Definitions/Concepts/Ideas
Principles of agile methods	Agile Manifesto (Agile Manifesto 2001)	• The best architectures, requirements, and designs emerge from self-organising teams • An adaptive short-term plan that evolves during software development provides more flexibility in responding to change and it is considered more effective than detailed "up-front" project plan with heavy documentation
	ASD (Highsmith 2000, 2009)	• Agility balances structure and flexibility, dependence and autonomy. Some structure, adequate documentation and some up-front architecture work, but not too much • Over the lifecycle of the project, architecture evolves with every iteration • Social architecture enables organisations and teams to face the volatility of their environment. It produces the best and the most innovative products • Architects need to be heavily involved in project integration
	Crystal method (Cockburn 2002, 2004, 2007)	• The system architecture evolves and also handles technology and business requirements changes over time • There is disagreement between developers about how much architecture to develop early in the project • Simple architectures are reasonably straightforward to upgrade to their next stage of complexity and performance • In large projects, the giant architecture should be broken up into smaller pieces that can be built and tested incrementally • Architecture description is text and/or drawings showing the major components and interfaces of the system
	DSDM (Stapleton 1997)	• It is important to understand not only the functionality to be deployed but also the system architecture that will be used • Architecture definition is the first system architecture sketch, and it is allowed to change during the course of the DSDM project
	Scrum (Schwaber 2004; Schwaber & Beedle 2002)	• Product Backlog represents a common repository • The high-level design of the system including the architecture is planned based on the current items in the Product Backlog • The architecture and the design of the system evolve during the Sprint development
	XP (Beck 2000)	• Architecture doesn't necessarily push the system into any sense of cohesion. It is a big box and connections • Parts of the architecture are captured by the system metaphor, which ensures that everyone in the team can tell about how the system as a whole works
Agile development literature	Ali Babar et al. 2009	• Using the architectural description helps the new team's members to be successfully integrated in the team

	Ambler 2014b	• Agile EA should be a team effort following the strategy of "everyone owns the architecture" where big up-front design is not required and a minimum documentation is required
	Batra 2009	• The principle, “the best architectures, requirements, and designs emerge from self-organising teams,” may need to be modified, especially if the size of the project is large. Large projects will need some degree of hierarchical structure to ensure accountability
	Boehm & Turner 2003	• Agile methods can be difficult to scale up to large projects because of a lack of sufficient architecture planning and over focusing on early result
	Bosch & Bosch-Sijtsema 2010	• The architecture is an important communication tool and a coordination mechanism in GDAD • Without using agile EA, road mapping, product management and integration in GDAD are done through meetings that require travelling or attending teleconferences outside work hours, which need big amount of effort and communication, high integration cost, and high coordination and communication cost
	Buckl et al. 2011	• Agile EA faces four challenges; stakeholder's satisfaction, customer's requirements, stakeholders' commitment, and the flexibility to requirement changes that can’t be addressed by any of the traditional EA frameworks • Agile EA helps in receiving early feedback to address any change needed and maintain early decisions and clear focus of agile team
	Gill 2013a	• Agile EA describes the overall structural, behavioural, social, technological, and facility elements of an enterprise • To address the design issue, agile EA has to be introduced or shared among GDAD teams, at least, at the minimum level
	Jaanu et al. 2012	• The lack of appropriate architecture decreases the knowledge sharing and communication effectiveness among agile GSD teams
	Kornstädt & Sauer 2007	• The architecture description provides terms and concepts that serve as a common language for all GDAD teams, which enables clear communication and arrangements
	Sauer 2010	• A shared software architecture serves as blueprint for all activities in the development process and ties them together • Architecture provides a plan for task allocation, facilitates the cooperation, and enables continuous integration reaching across GDAD teams • The architecture has to be documented in a way that it is understandable by all participants and the up-to-date architecture documents have to be accessible from all distributed sites
	Martini et al.	• The lack of appropriate architecture leads to

	2013	misunderstanding or an unnecessary flow of communication due to the insufficient definition of a system and software structure • Using reference architecture can increase the effectiveness of communication
	Madison 2010	• Agile EA helps in getting agile team members on board (on the track) through providing them with the holistic architectural direction setting • Using architecture is perceived as delivering large volumes of rich information in GDAD communication

Agile EA thus may facilitate communication by improving comprehensive vision through one common object of work that all GDAD participants use and understand (Bosch & Bosch-Sijtsema 2010), which increases knowledge sharing (Jaanu et al. 2012), and provides better understanding (Martini et al. 2013). The architecture description provides terms and concepts that serve as a common language for all DAD teams, which enables clear communication and arrangements (Beck 2000; Kornstädt & Sauer 2007). Without agile EA, road mapping, product management and integration are done through many meetings that require travelling to headquarters locations or attending teleconferences outside work hours, in most cases (Bosch & Bosch-Sijtsema 2010), which needs:

- big amount of effort and communication,
- high integration cost between DAD teams,
- high coordination and communication cost between GDAD developers,
- unintended resource allocation, which increases the coordination cost,
- insufficient pre-iteration (up-front design) and interface specification, which leads to late delivery or manual tests, and
- interaction between GDAD teams is more challenging due to different time-zones, cultures, languages, and work practices. This needs asynchronous communication or travelling to meet face-to-face with other teams or developers.

Avritzer et al. (2010) developed a “system of systems” process to solve project management problems that arise in the coordination of global software development projects. The authors found that (1) EA enhances communication among distributed

teams, (2) EA has the potential to guide task assignments and team coordination, (3) EA encourages and ensures that developers can identify the design rules and assigned tasks, (4) social architecture can be used to track the long-term project's communication, and (5) EA is more helpful to less experienced developers (Avritzer et al. 2010).

Design is a big issue in GDAD environment (Cockburn 2007; Hammad et al. 2014) as it could have negative effect on GDAD communication due to longer time needed and less efficient communication as a result of less understanding of design among GDAD teams (Cockburn 2007; Hammad et al. 2014). To address the design issue, agile EA has to be introduced or shared among GDAD teams at least for the minimum level (Gill 2013a; Cockburn 2007). In other words, to address the GDAD communication issue, the design issue needs to be addressed; and to address the design issue, agile EA artefacts need to be developed, evolved during software development iterations (Schwaber 2004), and used among the GDAD teams. It is anticipated that not only GDAD communication will be enhanced by using the agile EA, but also the overall agility and productivity will be increased (Alzoubi et al. 2015). In addition, although ASD does not prefer documentation, agile EA should be documented to a minimum level that can be understood by team members (Sauer 2010).

4.5.2 GDAD Active Communication

Communication has been defined as "a process in which participants create and share information with one another in order to reach a mutual understanding" (Rogers 1986, p. 199). Dennis et al. (2008) define communication as the development of shared understanding which is composed of two primary processes, conveyance of information and convergence on meaning (p. 576). Communication also refers to the process of exchanging information between senders and receivers (McQuail 1987). Communication can be defined as the sending message or transmission of data or information from sender to receiver (DeVito 1988). Miller (1976) claims that, "…communication has its central interest in those behavioural situations in which a source transmits a message to a receiver(s) with conscious intent to affect the latter's behaviour" (1976, p.92).

The definition of communication is quite different between the scholars and agile development practitioners (Branigan et al. 2010). Communication definitions, in general, draw our attention to the importance of sharing information (i.e. communication effectiveness). However, agile methods promise faster development, which means improving the communication efficiency too (Pikkarainen et al. 2008). Thus, agile methods require effective and efficient communication among stakeholders (i.e. users and customers) to achieve the highest software quality and customer satisfaction (Pikkarainen et al. 2008). Agility, the core of agile development, identifies how the agile team should communicate and respond to requirement's changes. It is clear from the agility definitions (as discussed in Section 2.5.1) that agile team members need to communicate efficiently and effectively. Therefore, agile methods require efficient and effective communication among team members and customers to achieve the highest software quality and customer satisfaction (Agile Manifesto 2001; Korkala et al. 2010; Pikkarainen et al. 2008).

As shown in Table 4.6, communication between developers and with customers is the core of agile development (Agile Manifesto 2001). Agile software development approaches have been introduced as alternative methods to the traditional "heavyweight" methods that have not had enough ability to address the current issues such as development time, cost, and respond to uncertain changeable customer's requirements (Beck 2000; Cockburn 2007; Highsmith 2000). To overcome these issues, agile development focuses on the role of people and communication. It values people and interactions over processes and tools, and customer collaboration over contract negotiation (Agile Manifesto 2001). It promotes close collaboration and communication between empowered development teams and customers (Agile Manifesto 2001).

Table 4.6: Principles of agile methods regarding GDAD active communication

Literature	Relevant Definitions/Concepts/Ideas
Agile Manifesto (Agile Manifesto 2001)	• The most efficient and effective method of conveying information to and within a development team is face-to-face conversation • Communication and collaboration have more value than tools and detailed documentation-based communication • Business people and developers must work together daily throughout the project
ASD (Highsmith 2000, 2009)	• In small team, using informal communication helps members to know when to develop a component • As project size and geographic team dispersion increase, informal communication needs to be formalised • Bigger teams involve more communication, more documents, and more politics • Co-located teams use lower effectiveness communications modalities since they have a good relationship context for sharing information. However, distributed teams need higher effectiveness communications modalities • Documents are not unimportant. They are just less important than working versions of the product
Crystal method (Cockburn 2002, 2004, 2007)	• Agile development can be characterised as a cooperative game of communication • Agility is a light human and communication-oriented, but with sufficient rules • Large projects with distributed teams, require more coordination and communication than small co-located teams • In GDAD, individuals must have open communication channels between each other. Communication technology may lead to a tremendous loss in communication efficiency, which means losing time and increasing mistakes • GDAD communication can result in less cost and errors by making it easier and with less coordination • Most projects require documentation in the least form for effective communication • When all communication is online, visible to everyone, there is no rumours grow in hiding, and there is only "us"
DSDM (Stapleton 1997)	• Developers should be able to communicate their ideas in non-technical language • Prototyping increases the communication effectiveness between the developers and with the stakeholders • The inadequate documentation and the lack of formal communication result in unnecessary revisits to areas already agreed
Scrum (Schwaber 2004; Schwaber & Beedle 2002)	• Scrum Master enhance coordination by communicating and working with management, team and customer • Scrum meetings (daily, Sprint) enhance communication among team • Team dispersion minimises human interaction and face-to-face communication, which demands written communication to minimise misunderstandings and errors
XP (Beck 2000)	• Communication and collaboration among team members support agile development • Frequent feedback plays an important role in attaining easier communication • Pair programming enhancing communication as a continuous verbal communication

GDAD communication can take place using various communication channels, such as face-to-face, voice-only, text, and so on. Each channel can transmit varying amounts of information and each of which requires a varying amount of interaction and attention from the recipient of the message (Daft et al. 1987; Dennis et al. 2008). Each channel may not only provide information about the message itself, but also include subsidiary data that may or may not be helpful to the task at hand (Helquist et al. 2011). For example, a face-to-face environment provides a "richer" communication channel (Daft et al. 1987) that includes not only verbal content which carries the main message, but also nonverbal content including gestures and body that may be helpful in understanding the intended message (Daft et al. 1987; Helquist et al. 2011). Rich media is preferred for communication that is high in equivocality, while media low in richness is favoured for communication that is less equivocal (Daft et al. 1987). Each communication channel has benefits and drawbacks according to the context being applied (Helquist et al. 2011).

Shared understanding is constrained by contextual or environmental factors as well as group characteristics such as proximity and the ability to communicate synchronously (i.e. two-way communication occurs at the same time) (Helquist et al. 2011). GDAD teams that communicate asynchronously (i.e. two-way communication does not occur at the same time) often find it more difficult to coordinate actions than proximally located teams (Dennis et al. 2008; Helquist et al. 2011). In order for GDAD teams to efficiently and effectively communicate, they must utilise appropriate communication channels (Korkala & Maurer 2014). GDAD teams use and adapt various tools and channels based on the fit between communication requirements and the task at hand (Gill & Bunker 2013; Green et al. 2010; Helquist et al. 2011). Highsmith and Cockburn (2001) reported that teams can be more effective in responding to changes if they can reduce the cost of sharing information between people, and reduce the elapsed time between making a decision and understanding the consequences of that decision (Highsmith & Cockburn 2001). We do not argue that any one medium is better than another. We argue that most "tasks are composed of a series of communication processes that need different media capabilities. For all but the simplest tasks, communication performance will be enhanced when different media are used at different times; it is usually best to use several media either simultaneously (e.g., face-to-face communication accompanied by

documents; telephone conferencing with synchronous electronic conferencing) or in succession (e.g., conveying information via e-mail first, followed by conversing over the phone). Additionally, we propose that as the familiarity with the task, individuals, and communication media increases, the need for media supporting high synchronicity is reduced" (Dennis et al. 2008, p. 576).

Communication in software development can be divided in two general types; informal and formal communication (Herbsleb & Mockus 2003). Informal communication refers to personal peer-oriented conversation among developers which occurs outside the official structure and sometimes without the knowledge of management (Herbsleb & Mockus 2003). Informal communication in ASD is more important than formal communication because informal communication helps in addressing changes in requirements quickly (Jaanu et al. 2012). Formal communication refers to the explicit clear communication such as the agile requirements backlog, plans and card walls etc. (Herbsleb & Mockus 2003). While ASD prefers informal communication over the formal communication in co-located teams, formal communication could be of high importance in GDAD environment (Gill et al. 2012). Whether the communication is formal or informal, there is a need to understand the two important dimensions of active communication: communication efficiency and effectiveness (Dorairaj et al. 2011, Korkala et al. 2010; Pikkarainen et al. 2008). Active communication is considered vital in co-located agile development teams to overcome the uncertainty and changeable customer requirements. Active communication is more important in GDAD because there are fewer opportunities for informal face-to-face communication and there is a need to meet the many challenges, as discussed in Ch.3 (Holmstrom et al. 2006b; Shrivastava & Rathod 2015).

As shown in Tables (4.6, 4.7, 4.8), on the one hand, active communication is at the heart of agile development principles and practices. Agile development approaches promote active communication among the team's members and with customers through continuous and face-to-face informal communication, using minimum documentation and formal communication, but sufficient rules. Moreover, agile EA has been realised by the prior literature as an important principle for improving active communication in large projects. On other hand, prior literature provides various definitions and

descriptions of active communication. While the literature is vague about the underlying dimensions and measures of active communication, there is a common theme underlying the various definitions and descriptions in that active communication is generally defined in terms of sharing and understanding information (e.g., Balijepally et al. 2009; Misra et al. 2009; Pikkarainen et al. 2008).

Moreover, the prior literature tends to view active communication as consisting of two important scopes that correspond to our conceptualisation of the two dimensions of GDAD active communication: communication efficiency and communication effectiveness. Communication efficiency and communication effectiveness have been invariably viewed by prior ASD literature; however, none of the previous research has given a clear definition for either communication efficiency or communication effectiveness. In fact, the previous literature was confusing about the use and differences between communication efficiency and communication effectiveness. In the next sections (i.e. Section 4.5.2.1 and Section 4.5.2.2) each dimension will be defined, since the definitions of them are not clear and scarce.

4.5.2.1 Communication Efficiency

According to the Oxford Dictionary, efficient "applies to someone or something able to produce results with the minimum expense or effort". Efficiency is concerned with short manufacturing times, lead times, cycle times and work times (Franke et al. 2010). Efficiency relates to time, cost, resources, or effort associated with communication (Lee & Xia 2010). It also refers to doing things (i.e. any task) right, even if it is not important to the job (i.e. the task is completed meeting all the standards of time, quality, etc.) (Frøkjær et al. 2000; Melo et al. 2011). Communication efficiency refers to sharing information in a timely manner (Clarke & Brennan 1991; Herbsleb & Mockus 2003). Accordingly, communication efficiency can be defined as delivering a message to a receiver with high quality and with minimal time, cost, effort, and resources required to establish communication.

Table 4.7 summarises the previous literature body on agile communication efficiency. There is no doubt that geographical distance and temporal distance are the main reasons of decreasing the efficiency of GDAD communication (Hummel et al. 2015; Misra et al.

2009). Although face-to-face is the most efficient channel of communication (Fruhling & De Vreede 2006; Pikkarainen et al. 2008), GDAD communication can be efficient by communicating timely and with the right people from the other teams (Sindhgatta et al. 2011), and by employing a Scrum role that would facilitate the communication between distributed teams (Valimaki & Kaariainen 2008). According to Sarker and Sarker (2009), agility in GDAD is all about how efficiently teams communicate and respond to challenges.

Table 4.7: Literature review about GDAD communication efficiency

Literature	Relevant Definitions/Concepts/Ideas
Fruhling & De Vreede 2006	• Refactoring and simple design influence communication by enabling simplicity, which is required to communicate efficiently
Hummel et al. 2015	• Direct communication has a positive impact on communication efficiency, whereas more indirect communication may inhibit communication efficiency
Korkala et al. 2010	• Agile practices are most successful in the environments where rapid communication is enabled • Using interactive media for efficient communication can be very difficult in GDAD due to temporal distance
Korkala & Abrahamsson 2007	• The lack of efficient communication can cause severe problems even in small ASD projects
Pikkarainen et al. 2008	• Face-to-face is the most efficient communication channel • Efficient communication is achieved by changing to informal communication using such practices as planning games, project reviews, stand-up meetings, pair programming, small releases, Sprint planning, and continuous integration • The open office space is an efficient way of improving informal communication as it provides more efficient discussion and solution forum of the project goals and status, on daily basis
Misra et al. 2009	• Rapid communication is a success factor of GDAD projects • Larger team might pose great hindrance to fast communication
Sindhgatta et al. 2011	• Splitting work across sites slows the work down • Communication efficiency can be enhanced through timely communication and communicating with the right people
Valimaki & Kaariainen 2008	• A person (Scrum role) can communicate efficiently with another Scrum role in distributed sites and inside his/her own site

4.5.2.2 Communication Effectiveness

According to the Oxford Dictionary, effective "describes something which successfully produces an intended result, without reference to morality, economy of effort, or efficient use of resources". Effectiveness is the accuracy and completeness with which

users achieve certain goals (Frøkjær et al. 2000). It also refers to doing the right things just to the tasks which are important to the job, even if they are completed without meeting standards of time, quality, etc. (Franke et al. 2010; Melo et al. 2011). Communication effectiveness means as little as possible disruption, minimal waiting time to get the required information and minimal chances of misunderstanding (Clarke & Brennan 1991; Dennis et al. 2008; Herbsleb & Mockus 2003; Ko et al. 2005; Rogers 1986). Accordingly, communication effectiveness can be defined as delivering a message to the receiver who understands it as it was intended with minimal disruption and misunderstanding, even if it takes a long time.

To achieve the effectiveness of communication, "the richness of the medium should match the level of message ambiguity. When the communication concerns well-defined issues and information, equivocality is low. Precise written and quantified data can be communicated through media low on the richness hierarchy. On the other hand, highly equivocal messages demand rich media to facilitate understanding and the emergence of a common perspective and understanding." (Daft et al. 1987, p. 359). Equivocality refers to confusion, disagreement and lack of understanding (Daft et al. 1987).

Table 4.8 summarises the previous literature body on agile communication effectiveness. While literature on communication efficiency is limited, literature on communication effectiveness is much more limited. In general, studies refer to active communication as communication effectiveness only. In other words, they assume that communication is only the effective communication. In plan-driven development, "effective communication has been offered as a solution to many of the problems encountered in the CIS development process, especially in the requirements definition phase" (Bostrom 1989, p. 280). This is also true for agile development.

Table 4.8: Literature review about GDAD communication effectiveness

Literature	Relevant Definitions/Concepts/Ideas
Ambler 2014a	• Effective communications realise that the goal is to share information in two-way street, typically
Balijepally et al. 2009	• Pair programming increases communication effectiveness through fostering developers' communication skills and improving interactions between them
Batra et al. 2011	• Providing structure by a coordinator (process facilitation) achieve significantly better communication effectiveness in GDAD
Cannizzo et al. 2008	• Communication effectiveness means minimal disruption, waiting time, and misunderstanding to get the required information • Communication effectiveness requires flexibility within the team in terms of working hours and immediate feedback which reduces waiting time, clashes, and helps in addressing problems
Dorairaj et al. 2011	• Communication effectiveness facilitates knowledge transfer rapidly between team members, allows team members to understand the requirements from clients, and helps team members perform development activities efficiently • Communication effectiveness can be increased in GDAD by reducing time-zone differences between teams, leveraging communication tools and techniques, addressing language barrier, developing trusted relationships among team members, and increasing the frequency of formal and informal communication
Green et al. 2010	• The use of agile method together with effective communication tools (e.g., teleconference) result in high quality GDAD systems
Holmström et al. 2006b	• GDAD requires effective communication and instant feedback between teams and from the customer
Hummel et al. 2015	• Effective communication is achieved by such agile practices as pair programming, Sprint planning, Sprint review, and so on
Layman et al. 2006	• Communication effectiveness can be enhanced in globally-distributed XP projects by encouraging developers to work closely with project management teams on a daily basis, assigning an individual to play the role of the customer up front, and using the available synchronous communication tools
Lee & Yong 2010	• Effective communication (e.g., clear communication) helps build a foundation of trust and respect of cultural differences, and decreases the effect differences in languages in GDAD
Ramesh et al. 2012	• Increasing project monitoring and control between remote team members lead to better communication effectiveness in GDAD
Melnik et al. 2006	• Automated acceptance tests (i.e. unit testing and automated accepted test) are effective communication media for customers and developers to exchange business requirements, domain knowledge, and the implementation status
Paasivaara & Lassenius 2014	• Scrum practices enable higher communication effectiveness in GDAD as they provide a frame for communication that reminds team members to interact closely and regularly
Pikkarainen et al. 2008	• Face-to-face is the most effective communication as a way to discuss and facilitate design problem solving

Sharp & Robinson 2007	• System metaphor serves as a communication platform with a common vocabulary that is stated to improve mutual understanding and communication effectiveness
Vidgen & Wang 2009	• The co-located office and the on-site customer serve as effective communication channels
Yadav et al. 2007	• Project communication planning, even if it is not very detailed in nature, has positive influence on communication effectiveness

Again, as for communication efficiency, face-to-face is still the best channel to achieve effective communication (e.g., Pikkarainen et al. 2008; Vidgen & Wang 2009). Agile practices such as Scrum Master role, Sprint, pair programming and so on, are seen as means for achieving effective communication in both co-located and GDAD environment (e.g., Balijepally et al. 2009; Hummel et al. 2015; Layman et al. 2006)". Moreover, using structure, control, and planning are assumed to enhance effective communication in both co-located and GDAD environment also (e.g., Batra et al. 2011; Ramesh et al. 2012; Sharp & Robinson 2007; Yadav et al. 2007). In addition, access to appropriate means of communication that match the situational needs of the user rather than choosing one single communication media was found a key factor for GDAD effective communication (Boden et al. 2007; Gill & Bunker 2013; Green et al. 2010; Valimaki & Kaariainen 2008). Another key factor of GDAD effective communication is the use of a central digital repository for the story boards, which is accessed by all team members (Gill & Bunker 2013; Maruping 2010; Valimaki & Kaariainen 2008).

4.5.3 GDAD Performance

Researchers have diverse interpretations of software development performance. Some have referred to it as a project success (e.g., Mahaney & Lederer 2006; Misra et al. 2009). Project is assumed successful if it is completed within or close to the success criteria boundary such as the estimated time/schedule, budget/cost, scope (functionality) and acceptable level of quality (Mahaney & Lederer 2006). Time, budget and quality are the key components of any project's success (Misra et al. 2009). Moreover, Drury-Grogan (2014) refers to project performance as project success and distinguish between project success and project management success. Project success "is measured against a project's overall achievement of the project's objectives, whereas project management success is measured using the traditional and oft-used measures of time (schedule), cost

(budget) and quality" (Drury-Grogan 2014, p. 507). However, researchers argue that it is impossible to distinguish between project success and project management success as these two concepts actually overlap (Drury-Grogan 2014).

Others refer to performance as project effectiveness (e.g., Dyba et al. 2007; Jiang & Klein 2000). Project is assumed to be effective if it meets the speed, schedule and efficiency (Aladwani 2002; Jiang & Klein 2000). Aspects related to effectiveness are project duration, effort and quality (Dyba et al. 2007). Wallace et al. (2004) define two types of performances: product performance and process performance. While product performance is a measure of the success of the system developed during the development project, process performance refers to success of the development process (i.e., the extent to which the project was delivered on schedule and within budget) itself (Wallace et al. 2004).

In general, both agile development literature and traditional software development literature have looked at software development performance through three dimensions: functionality, on-time completion, and on-budget completion (Aladwani 2002; Lee & Xia 2010). However, we argue that quality is an important dimension of software development performance (Chow & Cao 2008; Drury-Grogan 2014; Dyba et al. 2007; Misra et al. 2009). Therefore, this study refers to on-time completion, on-budget completion, functionality and quality as the four performance dimensions (see Table 4.9). On-time completion refers to the extent to which a software project meets its baseline goals for duration (Lee & Xia 2010). On-budget completion refers to the extent to which a software project meets its baseline goals for cost (Aladwani 2002; Lee & Xia 2010). Functionality refers to the extent to which the delivered software project meets its functional scope goals, user needs, and technical requirements (Aladwani 2002; Lee & Xia 2010). ISO 9126, the standard for evaluating software quality, defines quality as "the totality of features and characteristics of a product or service that bears on its ability to satisfy given needs" (Al-Kilidar et al. 2005). Also, quality refers to delivering a good working product that satisfies the requirements stipulated in the problem statement and produces correct results (Balijepally et al. 2009; Chow & Cao 2008). Satisfaction "represents affective response of the individual to the overall task" (Balijepally et al. 2009, p. 102).

Table 4.9: Software performance aspects

Performance Aspect	Literature	Relevant Definitions/Concepts/Ideas
On-Time Completion	Chow & Cao 2008	• Refers to delivering software project on time
	Drury-Grogan 2014	• Refers to the scheduling of tasks and completion dates
	Lee & Xia 2010	• The extent to which a software project meets its baseline goals for duration
	Melo et al. 2011	• Accounts for meeting datelines, overtime needed to complete the work, and other time related issues
On-Budget Completion	Chow & Cao 2008	• Delivering software project within estimated cost and effort
	Drury-Grogan 2014	• Refers to the overall costing of the project
	Lee & Xia 2010	• The extent to which a software project meets its baseline goals for cost
	Mahaney & Lederer 2006	• The extent to which a software project is completed within or near the estimated budget
Software Functionality	Chow & Cao 2008	• Meeting all requirements and objectives
	Lee & Xia 2010	• The extent to which the delivered software system meets its functional goals, user needs, and technical requirements
	Mahaney & Lederer 2006	• The extent to which a software project meets its technical goals
Software Quality	Chow & Cao 2008	• Refers to delivering good product or project outcome
	Conboy & Fitzgerald 2004	• Refers to achieving high standards of the software and supporting documentation produced, and the development team
	Drury-Grogan 2014	• Refers to how well the finished product functions
	Mahaney & Lederer 2006	• The extent to which the project performance is improved
	Misra et al. 2009	• Productivity, customer satisfaction, business processes, and functionality can be perceived as quality criteria

4.6 Hypotheses Definitions

Hypothesis describes a positive or a negative relationship between a dependent construct and an independent construct. The hypotheses are reflected in the research model as arrowed lines which start from the influential (independent) construct and end in the dependent construct.

Table 4.10: Summary of the research hypotheses

No.	Hypothesis	Relationships between the Constructs	Direction of the Relationship
1	Hypothesis 1a (H1a)	Agile Enterprise Architecture positively affects the efficiency of the GDAD communication	+ Agile EA → Communication efficiency
2	Hypothesis 1b (H1b)	Agile Enterprise Architecture positively affects effectiveness of the GDAD communication	+ Agile EA → Communication effectiveness
3	Hypothesis 1c (H1c)	Agile Enterprise Architecture positively influences on-time completion of GDAD software	+ Agile EA → On-Time completion
4	Hypothesis 1d (H1d)	Agile Enterprise Architecture positively influences on-budget completion of GDAD software	+ Agile EA → On-Budget completion
5	Hypothesis 1e (H1e)	Agile Enterprise Architecture positively influences GDAD software quality	+ Agile EA → Software quality
6	Hypothesis 1f (H1f)	Agile Enterprise Architecture positively influences GDAD software functionality	+ Agile EA → Software functionality
7	Hypothesis 2 (H2)	GDAD communication efficiency negatively affects effectiveness of the GDAD communication	- Communication Efficiency → Communication effectiveness
8	Hypothesis 3a (H3a)	Communication efficiency positively influences on-time completion of GDAD software	+ Communication Efficiency → On-Time completion
9	Hypothesis 3b (H3b)	Communication efficiency positively influences on-budget completion of GDAD software	+ Communication Efficiency → On-Budget completion
10	Hypothesis 3c (H3c)	Communication efficiency positively influences GDAD software functionality	+ Communication Efficiency → Software functionality
11	Hypothesis 3d (H3d)	Communication efficiency positively influences GDAD software quality	+ Communication Efficiency → Software quality
12	Hypothesis 4a (H4a)	Communication effectiveness negatively influences on-time completion of GDAD software	- Communication Effectiveness → On-Time completion
13	Hypothesis 4b (H4b)	Communication effectiveness negatively influences on-budget completion of GDAD software	- Communication Effectiveness → On-Budget completion
14	Hypothesis 4b (H4c)	Communication effectiveness positively influences GDAD software functionality	+ Communication Effectiveness → Software functionality
15	Hypothesis 4d (H4d)	Communication effectiveness positively influences GDAD software quality	+ Communication Effectiveness → Software quality

As shown in Figure 4.5, the research model in this study has four major hypotheses. Hypothesis 1 posits that agile EA has positive effects on GDAD communication efficiency and communication effectiveness, and on GDAD performance. Hypothesis 2 posits a trade-off relationship between GDAD communication efficiency and effectiveness. Finally, hypotheses 3 and 4 posit that GDAD communication efficiency and communication effectiveness have differential effects on the four dimensions of GDAD performance: on-time completion, on-budget completion, software functionality, and software quality. The following sub-sections discuss how each hypothesis (sub-hypotheses) has been derived. Table 4.10 provides a summary of the research hypotheses.

4.6.1 Effect of Agile EA on GDAD Active Communication

While communication is a process that is closely tied to physical proximity, agile EA can be seen as an outcome of that process (Bernard 2012; Tamm et al. 2011). Agile EA as a whole evolves which means that up-to-date real-time information about integrated business and software architecture elements are shared between GDAD teams. This provides a holistic shared view to GDAD teams as the different parts of it (e.g. business and IT capabilities, services and products) are developed and delivered by the GDAD teams in different increments at different times. Bass et al. (2013) state that "software architecture represents a common abstraction of a system that most, if not all, of the system's stakeholders can use as a basis for creating mutual understanding, negotiating, forming consensus, and communicating with each other" (p. 29). Therefore, this shared view will increase the common ground among distributed GDAD teams (Niemi & Pekkola 2015). High common ground (1) reduces uncertainty just as frequent communication does (Tamm et al. 2011), (2) produces positive attributions when real data are absent (Bernard 2012), (3) increases the awareness, which increases the coordination among GDAD teams (Bernard 2012), and (4) facilitates information sharing by providing common data definitions and structures (Boh & Yellin 2006).

On one hand, high common ground using the agile EA shared view is associated with reduced cognitive effort to encode and decode messages (Dennis et al. 2008), yielding faster message transmission, so a message can be assessed and modified quickly (Clark 1996; Ko et al. 2005). So, "in situations where individuals have shared mental models,

encoding and decoding familiar information should be faster" (Dennis et al. 2008, p. 583). It can also provide team members with the ability to receive immediate feedback (Clark & Brennan 1991). On the other hand, low common ground may impair development of understanding because members will not have the time required to fully process the information, which may cause a greater cognitive load on the individual (Te'eni 2001) and encourage premature action (Dennis et al. 2008). Therefore, we propose

H1a: Agile Enterprise Architecture positively affects the efficiency of the GDAD communication.

Moreover, this shared view may serve as a common information model for enabling rich communication among GDAD teams and can provide a single view of the agile EA information to GDAD stakeholders (Ambler 2014; Gill 2015b; Madison 2010; Svensson et al. 2012). It also provides terms and concepts that serve as a common language for all GDAD teams, which enables clear communication (Kornstadt & Sauer 2007). It decreases the misunderstanding or an unnecessary flow of communication due to the insufficient definition of a system and software structure (Martini et al. 2013). It helps GDAD teams to coordinate their work through interfaces of their components such that each component can be developed separately, which means that the frequencies of communication as well as considering development of other components are decreased (Bernard 2012; Tamm et al. 2011), and the need for synchronous communication is reduced (Dennis et al. 2008). To sum up, this shared view, which provides simplicity as well as ease of development and incorporation with external developers, can be a mechanism for communication and coordination in GDAD and may provide the same efficiencies as co-located development (Bosch & Bosch-Sijtsema 2010; Jaanu et al. 2012). Therefore, we propose

H1b: Agile Enterprise Architecture positively affects effectiveness of the GDAD communication.

4.6.2 Effect of Agile EA on GDAD Performance

Software architecture in distributed projects is treated in the literature mainly from two points of view: to split the work into meaningful tasks that can be implemented by distributed teams and as common artefact that allows the teams to coordinate their work (Sauer 2010). The development of architecture should be seen as an iterative process, not as an up-front task that establishes unalterable structures (Sauer 2010). The architecture has to be documented in a way that it is understandable by all participants and the up-to-date architecture documents have to be accessible from all sites (Sauer 2010).

Agile EA draws from a uniform infrastructure, platform, application, and communicates the architecture value and status with all stakeholders (Madison 2010), which provides single sources (digital repository) with a better access to customer and shared data, more accurate and timely accessible data, better understanding of processes, and more manageable IT environment (Bernard 2012; Ross et al. 2006; Venkatesh et al. 2007). This shared architecture repository serves as a blueprint for all activities in the development process and ties them together, which provides a plan for task allocation, facilitates cooperation, and enables continuous integration reaching across GDAD teams and members (Sauer 2010). This repository, which holds the story boards that all team members have access to, allows all team members to view and manipulate the story boards in real-time (Maruping 2010).

Agile EA provides the basis for architecture rules, which improves implementation consistency and reduces the number of errors (Kornstädt & Sauer 2007). It also helps in measuring the architecture state of the product (code, functionality, refactoring) (Madison 2010). It improves resource integration and interoperability by improving performance, re-using of technology and expertise, and increasing IT responsiveness (Bernard 2012; Ross et al. 2006). Moreover, using agile EA helps to reduce the cost and eliminate the redundancy by reducing support and IT costs, reducing resource duplication, involving fewer people in a process, and reducing skill variation and focusing on core competencies through selective outsourcing (Ross et al. 2006; Tamm et al. 2011; Venkatesh et al. 2007). In addition, agile EA is the placeholder for software

quality, modifiability, security, and reliability (Bernard 2012; Kornstadt & Sauer 2007; Ross et al. 2006). Therefore, we propose

H1c: Agile Enterprise Architecture positively influences on-time completion of GDAD software.

H1d: Agile Enterprise Architecture positively influences on-budget completion of GDAD software.

H1e: Agile Enterprise Architecture positively influences GDAD software quality.

H1f: Agile Enterprise Architecture positively influences GDAD software functionality.

4.6.3 Relationship between GDAD Active Communication Dimensions (Efficiency, Effectiveness)

"The context in which communication occurs can have a significant effect on the need for particular types of communication processes" (Dennis et al. 2008, p. 589). Due to GDAD communication challenges, the message may not be received as it was effectively intended (Hinds & Mortensen 2005). The higher communication effectiveness comes at the price of considerably longer time (Dyba et al. 2007). Considering the impacts of time, cost and effort on communication, GDAD team tends to first decide what and how much they would communicate. This choice in turn affects communication effectiveness. The short message may be insufficient to deliver clear information (Clarke & Brennan 1991). Low communication efficiency can negatively impact communication effectiveness by increasing delays that impede the rapid development of shared understanding (Dennis et al. 2008).

Moreover, if the message is complex, with large amounts of information or high diversity of information, team member will require more time to assess and deliberate on the information (Dennis et al. 2008). Communication efficiency may impair development of understanding because members will not have the time required to fully process the information, which may cause a greater cognitive load on the individual (Te'eni 2001) and encourage premature action (Dennis et al. 2008). Accordingly, increasing the communication efficiency may come at the expense of communication effectiveness and vice versa. Based on the discussion in this section and the definition

of communication efficiency provided in Section 4.5.2.1 and the definition of communication effectiveness provided in Section 4.5.2.2, we propose

H2: GDAD communication efficiency negatively affects effectiveness of the GDAD communication.

4.6.4 Effect of Communication Efficiency on GDAD Performance

Communication efficiency is essential and facilitates the complex cognitive processing necessary to achieve quality solutions (Balijepally et al. 2009). It is important for GDAD teams to facilitate knowledge transfer efficiently which allows team members to understand the customer requirements and to help team members perform development activities efficiently (Dorairaj et al. 2011). Korkala et al. (2010) argue that GDAD practices are the most successful in the environments where efficient communication is enabled. Inefficient communication along with volatile requirements can cause severe problems for GDAD projects (Korkala & Abrahamsson 2007; Layman et al. 2006).

Efficient communication leads to fast responding to customer requirements, which results in high agile development performance (Cockburn 2007). Delay in identifying project impacts, dependencies and resultant changes in GDAD environment lead to longer development duration and extra cost (Boehm & Turner 2003). If the efficiency of GDAD communication is high, the amount of extra time and costs required for handling customer requirements changes is minimal (Cockburn 2007). This may decrease the additional time and cost, and meet the assigned time and budget targets (Lee & Xia 2010). Based on the discussion in this section and the definitions of communication efficiency and communication effectiveness provided in the above sections, we propose

H3a. Communication efficiency positively influences on-time completion of GDAD software.

H3b. Communication efficiency positively influences on-budget completion of GDAD software.

H3c. Communication efficiency positively influences GDAD software functionality.

H3d. Communication efficiency positively influences GDAD software quality.

4.6.5 Effect of Communication Effectiveness on GDAD Performance

The higher communication effectiveness comes at the price of considerably longer time and higher cost to deliver, while the shorter and faster communication come at a price of a noticeably lower software performance (Dyba et al. 2007). Bostrom (1989) states that ineffective communication "during any of the phases of system development is negatively correlated with system success" (p. 283).

Effective communication is crucial for GDAD to facilitate knowledge transfer clearly between team members, to allow team members to understand the requirements from clients and to help team members perform development activities efficiently (Dorairaj et al. 2011; Korkala & Maurer 2014). Effective communication technologies should be an integral part of the GDAD that can create high quality GDAD projects (Gill & Bunker 2013). To avoid possible communication inconsistencies and ambiguities, GDAD should have a "single knowledge repository" for managing communication among distributed teams (Gill & Bunker 2013). Using a combination of communication media (e.g., face-to-face, email, newsgroup) is more effective than choosing one single communication media (Green et al. 2010). Given the correct combination of communication methods and tools during the optimal phase of the software lifecycle, GDAD teams may be as effective as co-located agile teams (Gill & Bunker 2013; Green et al. 2010). Effective customer collaboration should be looked at as a key factor for successful GDAD projects (Korkala et al. 2010; Korkala & Abrahamsson 2007). Yadav et al. (2007) found that GDAD communication effectiveness mediates the relationship between project management (i.e. project communication, and project monitoring and control) and project success (i.e. project performance).

To effectively communicate about many different customer requirements and requirements' changes, a GDAD team may need new resources and capabilities or reconfigure existing resources and capabilities (Lee & Xia 2010). This requires a considerable amount of extra time and cost (Lee & Xia 2010). Furthermore, communication about customer's requirements and requirements' changes helps in correcting system configuration, and improve design and product quality (Bhalerao

2010). The functionality and quality of the software will not satisfy "up-to-date" customer needs if the team fails to embrace important changes (Lee & Xia 2010). Therefore, we propose

H4a. Communication effectiveness negatively influences on-time completion of GDAD software.

H4b. Communication effectiveness negatively influences on-budget completion of GDAD software.

H4c. Communication effectiveness positively influences GDAD software functionality.

H4d. Communication effectiveness positively influences GDAD software quality.

4.7 Chapter Summary

This chapter discussed the research model constructs and hypotheses. The research model was build based on the literature review (Ch.2, Ch.3, and Ch.4) and preliminary evaluation (Ch.5). This model includes three important constructs and relationships: agile EA, GDAD active communication, and GDAD performance. These constructs were rigorously identified from the previous literature and verified through preliminary evaluation. The major cooperated hypotheses are:

Hypothesis 1: Agile EA has positive effects on GDAD communication efficiency and communication effectiveness, and on GDAD performance.

Hypothesis 2: The relationship between GDAD communication efficiency and communication effectiveness is negative.

Hypotheses 3: GDAD communication efficiency positively affects the four dimensions of GDAD performance (on-time completion, on-budget completion, software functionality, and software quality).

Hypotheses 4: GDAD communication effectiveness negatively affects on-time completion and on-budget completion, and positively affects software functionality and software quality.

The following chapter discusses the research design, presenting the common paradigms and classification of IS research in order to define and classify the nature of this study so the best-fit method for this research can be chosen. It also discusses the research strategy, the data collection methods, and the analytical techniques used to empirically examine the proposed theoretical model and test the research hypotheses.

Chapter 5 Research Design and Methodology

In Ch.4, the research model constructs were discussed. The research model was rigorously developed based on the literature review in (Ch.2, Ch3, and Ch.4). The model involves three main constructs: agile EA, GDAD active communication (efficiency and effectiveness), and GDAD performance (on-time completion, on-budget completion, functionality, and quality). Moreover, the hypothesised relationships that link them with one another were discussed. Agile EA is hypothesised to have positive impacts on both communication and performance. Communication efficiency and communication effectiveness are negatively related to each other and have differential impacts on performance aspects. Having the research model and hypotheses established, now we can move to discuss the methodology of evaluating the model and testing the related hypotheses.

5.1 Introduction

This chapter, broadly, describes how the study was planned and implemented to fulfil the research objectives of the dissertation. It briefly describes the research design, research philosophy, research methods selection, research data collection and analysis plan as well as the research schedule for examining the constructs and their hypothesised relationships that structure the proposed model. This chapter discusses the issues that need to be addressed through answering the two central questions to the design of research: (1) what are the knowledge claims that have been made? (Including the theoretical perspective; positivism, interpretivism, and critical realism) and (2) what are the strategies, procedures, methods of data collection, and analysis methods that will be used to evaluate and validate the research model?

In Section 5.2, an overview of the research design is discussed, which is fundamentally based upon the aim and objectives of the study. Next, the research philosophy is

discussed in Section 5.3, which helps identify the philosophical standing as well as the underlying beliefs that direct and justify the research method used in this study. Section 5.4 gives a thorough description of the research method selection. The approaches towards a mixed survey and case study methods that aid in ensuring that the objectives of this study are adequately addressed are then discussed. The discussion focuses first on the preliminary model evaluation, which is discussed in Section 5.5. Survey instrument design and validation, and data collection and analysis are discussed in Section 5.6. The discussion continues with the case study collection and analysis used in this study, presented in Section 5.7. Section 5.8 specifies the human research ethics considerations. Finally, a brief summary and discussion of this chapter is presented in Section 5.9.

5.2 Research Design

A research design can be defined as the series of rationales and decisions that form strategies to generate valid and reliable research results by answering the research questions and testing the hypotheses (Cavana et al. 2001). A key principle to be followed to increase the reliability of the data presented in a case is the maintenance of a logical "chain of evidence" (Benbasat et al. 1987; Yin 2009). As Yin (2009) explained, "chain of evidence" represents the steps or the methodological procedures that allows following the derivation of any evidence from initial research questions to ultimate case study conclusions. Furthermore, it allows traceability to the research steps in either direction (from conclusions back to initial research questions or from questions to conclusions). This helps a case study to address the methodological problem of determining internal validity (Dube & Pare 2003).

This study is dominated by positivism and uses interpretivism philosophy. In a positivist setting, the research design includes the decisions about data collection methods, procedures of measurements and instrument generation, and population sample and data analysis procedures (Cavana et al. 2001). In interpretive settings, the research design includes the decisions about data collections from case studies or individuals and analysis procedures (Cavana et al. 2001).

This study used a combined approach of quantitative (survey) and qualitative (case study) to explain the study objectives. Using this mixed-method approach helps in addressing limitations for both qualitative and quantitative methods by providing the objectivity of the statistics and deeper understanding of the study context (Gable 1994; Kaplan & Duchon 1988). Quantitative approach enables the researcher to test the relationships between the variables of the research model, and provide evidence to support the research hypotheses (Creswell 2009; Gable 1994; Kaplan & Duchon 1988; Venkatesh et al. 2013). Qualitative approach enables better understanding of the variables and their relationships in the research model (Gable 1994; Venkatesh et al. 2013).

The research design is illustrated in Figure 5.1. The sequence of this research follows the recommendation of research process by Creswell (2009). The research process may start with a quantitative approach, where concepts and hypotheses can be tested, and then a qualitative approach can be used that involves detailed explanation of a few cases or individuals (Creswell 2009). This research includes five stages:

1. Identifying the research problem. In this research the problem was the issue of communication between distributed stakeholders in GDAD environment.

2. Reviewing the literature related to the research problem. In this research a systematic literature review was conducted to GDAD challenges, in general, and to GDAD communication challenges specifically (see Ch.3). This review enabled the conceptualisation of the important issues related to GDAD communication; challenges, techniques to overcome these challenges, and the impact of GDAD communication on GDAD performance. Moreover, this review was essential to develop the survey instrument.

3. Building the initial research model. This was done based on the literature review (stage 2) and the preliminary evaluation by some experts in the field; agile development practitioners and academics (Section 5.5).

4. Evaluating the research model. This was done by conducting survey and case studies. Firstly, the survey instrument was developed and evaluated to test the relationships between the variables of the research model (see Ch.6 and Ch.7). The findings of the survey were refined and elaborated by conducting 10 post hoc case studies using semi-structured interviews (see Ch.8).

5. Finally, interpretation of the findings through documentation. This is the last stage of this study, where all previous stages findings are documented in the thesis.

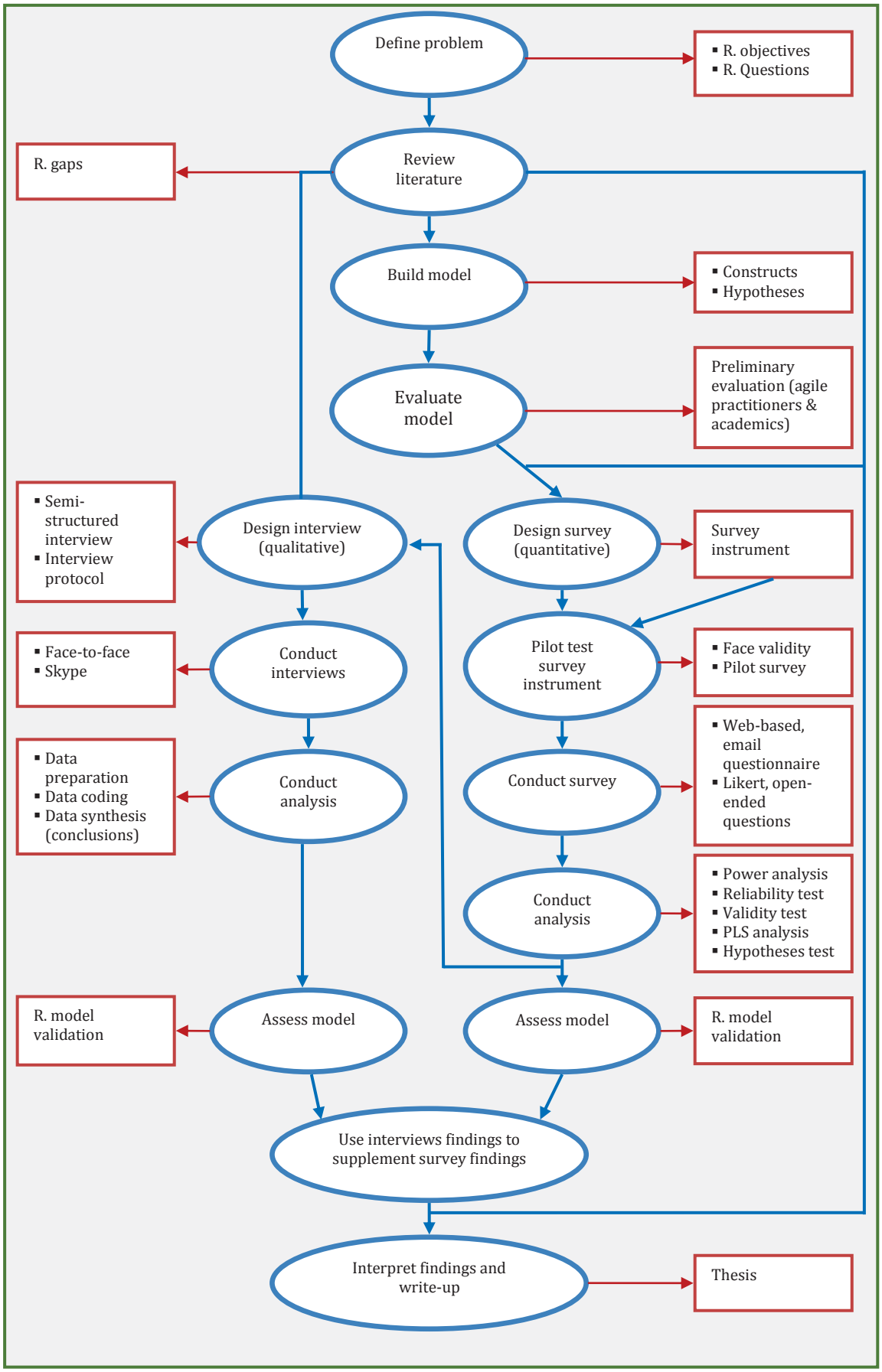


Figure 5.1. Research design

5.3 Research Philosophy

It is crucial for researchers to establish their philosophical standing before undertaking the detailed study (Neuman 2003). This is because the philosophy represents the critical examination of the grounds of the fundamental beliefs and the analysis of the basic concepts of these beliefs (Neuman 2003). The paradigm is a framework that guides the research throughout its different stages (Khazanchi & Munkvold 2003). The paradigm refers to the set of assumptions that frames the researcher's beliefs about the ontology, epistemology, ethics, and methodology of the research (Creswell 2009; Hirschheim & Klein 1989; Walsham 1995).

1. Ontology: ontology refers to researcher's assumptions about the nature of reality (what is believed to exist) about the research (Creswell 2009; Iivari et al. 1998). Ontology questions whether a real world exists and whether it depends on human thoughts and speeches or stands independently of human consciousness (Becker & Niehaves 2007).

2. Epistemology: epistemology refers to how knowledge about that reality is sought (can be obtained) (Iivari et al. 1998). Epistemology is the relationship between reality and researchers through answering the two questions; what is the nature of knowledge? And (2) how do we obtain valid knowledge? (Khazanchi & Munkvold 2003).

3. Methodology: methodology refers to methods, tools, procedures, techniques, and strategies that are chosen to seek and acquire that knowledge (Iivari et al. 1998). Methodology is the answer to the question of how researchers establish what they believe can be known (Guba & Lincoln 1994). The methodology will be discussed in detail in Section 5.3.2.

4. Ethics: ethics refers to the set of morals, values, and scientific principles that define the responsibility of researcher for the consequences of the research approach and

outcomes (Iivari et al. 1998). Section 5.8 discusses the ethical considerations for this study.

These assumptions play a vital role in the research through guiding the design and development process of the research (Babbie 2001). In the following sub-sections, the different research philosophical paradigms are discussed.

5.3.1 Defining Research Paradigms in IS Research

IS researches, traditionally, have adopted three common paradigms; positivism, interpretivism, and critical realism (Orlikowski & Baroudi 1991). Positivism and interpretivism are often mentioned as the dominant philosophy of IS research (Orlikowski & Baroudi 1991). The following sub-sections discuss these three paradigms. Table 5.1 summarises the basic characteristics of each of those three paradigms.

Table 5.1: Characteristics of Positivism, Interpretivism, and Critical Realism

	Research Perspectives		
Basic Belief	**Positivism**	**Interpretivism**	**Critical Realism**
Ontology	A single reality; knowable; probabilistic	Multiple realities; socially constructed	Reality can be only imperfectly and partially detained
Epistemology	Dispassionate; detached observer of truth	Subjective; values and knowledge emerge from the researcher-participant interaction	Knowledge is objectively constrained within a social context
Methodology	Observation; quantitative; statistical	Participation; Qualitative; Hermeneutical; Dialectical	Mixed-method are more suitable for this type of research
Ethics	Truth is universal and beautiful; prediction	Understanding is situated and with description	Control; prediction; understanding

5.3.1.1 Positivism Paradigm

The advocates of positivism paradigm argue that it represents the basis of natural and social researches. They argue that the social reality (e.g., beliefs, attitudes, or behaviours) is subject to universal laws that can be measured using traditional rigorous scientific methods (Orlikowski & Baroudi 1991). Applying random sampling,

researchers can predict the behaviours or characteristics of a certain population (Mingers 2001). In this paradigm, the researcher is an objective outsider, ideally. Positivist research is usually concerned with measuring a relationship between different constructs (variables) in the representative sample of the stated population (Shanks 2002). This measurement is usually done by testing hypotheses, quantifying measures of constructs, and deducing inferences about a phenomenon in that sample (Orlikowski & Baroudi 1991). Generally, positivist research is usually related to quantitative research approaches, which is dominated by survey method, and use statistical modelling for analysing the data (Cecez-Keananovic 2001). Despite the wide use in IS research, positivism has been subject to some criticisms, which are related to the underlying assumptions of this paradigm. For example, some authors argue that positivism doesn't have an established related theoretical basis to draw from, but rather uses other disciplines theories such as computer science and organisational behaviour (Dobson 2001). Moreover, positivism generally lacks the vision to intensely explain social reality, as reality is a complex phenomenon that can't be explained by statistical measurements (Kaplan & Duchon 1988).

5.3.1.2 Interpretivism Paradigm

In this paradigm, researchers attempt to understand the phenomenon through examining its context and through understanding the meanings that were assigned to it by people (Orlikowski & Baroudi 1991). They assume that reality is a social construct but it is subjective and based on shared meanings (Myers & Avison 2002). Interpretivists recognise both similarities and differences between the objects of social system (Walsham 1995). They mainly focus on investigating different perspectives of people. Therefore, this paradigm is highly recommended in the situation where the phenomenon is influenced by many factors or when several actors are included (Myers & Avison 2002). The complex issues related to the interaction between different IS applications and social phenomena such as individual behaviour or organisational practices drive the motivation of adopting interpretivism by many IS researchers (Klein & Myers 2002). Statistical tests may be not enough to understand a problem that exists in a social context (Babbie 2001). Therefore, interpretivists rely more on qualitative approaches than on quantitative approaches, which offer greater explanations and better understanding of the social phenomena (Putney et al. 1999).

5.3.1.3 Critical Realism Paradigm

Critical realism has attracted many researchers during the recent years due to the criticisms and limitations of both positivism and constructivism (Wynn & Williams 2012). The focus of researchers in this paradigm is to uncover the conflicts and contradictions within the social system (Orlikowski & Baroudi 1991). This paradigm helps researchers in attaining better understanding of human behaviours since it emphasises change and liberty of research (Wynn & Williams 2012). In general, critical theorists assume that people can consciously and deliberately act to change their economic and social conditions; however, their actions are limited by social and political dominations (Hirschheim & Klein 1989). It has been claimed that critical paradigm recognises the contributions of both positivism and interpretivism (Wynn & Williams 2012; Babbie 2001). This is because critical paradigm acknowledges the need for causal theory-based observations and interpretive descriptions that are based on the inter-subjective understanding of reality (Babbie 2001).

5.3.1.4 The Paradigm Adopted in this Study

The above research paradigms have been used in IS research individually or jointly. Some philosophical schools strongly recommend following only one of these paradigms in a single research study. Other schools believe that a combination of these paradigms can be used within different phases of a single study (e.g., Creswell 2009; Gable 1994; Kaplan & Duchon 1988; Venkatesh et al. 2013). However, using these paradigms in a research study has been criticised by a number of researchers due to the difficulties of considering their differences when put into practice, their institutional contexts influence on the researcher, and their impact on drawing the researcher's attentions to certain aspects rather than all aspects of the research (Gable 1994; Weber 2003).

The selection of ontological and epistemological position between the above research paradigms is not based on the judgment of which approach is superior. Instead, the selection of the research paradigm is based on the research study context, its purpose and its nature (Orlikowski & Baroudi 1991).

Positivism paradigm has been dominantly used in IS research (Orlikowski & Baroudi 1991). Chen and Hirschheim (2004) found that 81 percent of reviewed empirical research in eight major IS journals between 1991 and 2001 followed the positivism paradigm. This tendency continued in the following years (Wynn & Williams 2012). Following this tendency, this study is primarily guided by the positivist paradigm to address the research questions. In addition, based on the ontological, epistemological and methodological considerations, there are several rationales that justify this selection.

1. The nature of the derived research question: *How does agile EA affect GDAD active communication?* The question seeks to generalise the results that can be adopted by all GDAD organisations, not only those who use agile EA in their GDAD.

2. The fact that positivist studies are guided by clear guidelines that are described in detail with sample cases in the literature and describe how to address validity and reliability issues of the data.

3. From the ontological standpoint, positivists assume that object reality can be systematically and rationally investigated through empirical investigation and is driven by general causal law (Shanks 2002). This is relevant to this study, which intends to make inferences about the effect of agile EA on communication and project performance within the context of GDAD environment. The theoretical model is based on perspectives and theories drawn from the EA and communication literature. This characteristic is relevant to the epistemology of positivism.

4. This research study aims to empirically test the hypotheses proposed to shape the relationship between agile EA and GDAD communication. The researcher collects information using instruments based on measures completed by respondents. Hence, the researcher remains distant and does not directly interact with respondents (GDAD agile team members). The researcher merely interprets the results and has no influence on the data collected. This characteristic is also relevant to the epistemology of positivism.

5. The essential requirement of methodology in positivism paradigm is that general theories are used to generate propositions that are operationalised as hypotheses and subjected to replicable empirical testing. Hypotheses should be testable, and offer the opportunity for falsification or confirmation (Shanks 2002). In this study, constructs definitions and measures are based on existing constructs and are quantifiable. Some items were generated by the researcher based on expert's feedback. The hypotheses are generated from formal propositions. Moreover, the research follows the deductive reasoning for the purposes of confirming and extending previously stipulated hypotheses as a primary method of collecting evidence.

IS epistemology was argued to draw heavily from the social sciences since IS systems are social rather than technical systems (Hirschhiem 1985). However, the technology side is ignored in this view, which is not avoidable in the current IS systems (Burstein & Gregor 1999). Moreover, gaining knowledge about reality can be achieved by exploring the shared meanings about certain phenomenon (Klein & Myers 1999). In the context of this research, investigating the effect of using agile EA on the GDAD communication can be achieved through exploring the shared meanings of the stakeholders assigned to the issue of GDAD communication, which include the technical issues pertaining to this communication. It is strongly argued that a combination of paradigms would serve as the best option to accomplish all the objectives of this study. Accordingly, this study adopted interpretive paradigm, in addition to positivism paradigm, since it is completely justifiable under the interpretive paradigm to explore the shared meanings and the diversity of opinion regarding using agile EA to enhance communication in GDAD environment. Moreover, this study cannot consider itself to be of a critical nature since it is neither concerned with providing a social critique nor dictating an emphasis on change for any of the uncovered conflicts or disagreements within the phenomenon being investigated (Burstein & Gregor 1999).

5.3.2 Methodology

Research methodology involves the decisions and choices made concerning a research project. It also covers how research projects are planned and implemented, and how

data is collected and analysed (Yin 2009). The research methodology also defines the subset of acceptable research methods applied in an empirical study (Yin 2009). The following sub-sections shed light on the class of this study and the methods used to collect the data.

5.3.2.1 The Adopted Class of this Research Study

Neuman (2003) classified research studies into three general main types: explanatory, exploratory, and descriptive research. First, explanatory research tries to answer "why" and "how" questions; why things are the way they are and why something occurs, and how the relationships between two aspects of the phenomenon exist (Neuman 2003). A high level of control is involved in this type of research (Bernard & Bernard 2012). This control enables theory testing and explanation (Bernard & Bernard 2012). Usually, explanatory research attempts to extend a theory to new topics or issues, links these issues with general concepts and principles, and refutes or supports a prediction of the theory (Neuman 2003). Deductive methods of inquiry are predominant in explanatory research. This domination is because the research, in this type, tests (i.e. confirm or deny) an existing theory through analysing the collected data (Neuman 2003).

Second, exploratory research tries to answer the "what" question. It tries to relatively investigate a new topic or issue (s) in order to provide a new knowledge to the knowledge base about that topic, or provide better understanding about that topic (Babbie 2001). Exploratory research has been described as an approach to formulate more structured knowledge over time with the ability to create precise questions that will be better addressed in the future research (Neuman 2003). Exploratory research rarely provides conclusive answers to such new topics (Bernard & Bernard 2012). Exploratory research relies heavily on inductive methods approaches rather than relying on specific theory to draw the results (Neuman 2003).

Third, descriptive research tries to answer "who" and "how" questions about a social phenomenon. It aims to deliver high details about such a phenomenon, which helps in providing better explanation to that phenomenon structure as accurately as possible

(Babbie 2001). The output of this type, generally, aims at building new theory (Neuman 2003).

This study is of an explanatory nature due to: (1) the research is mainly interested in explaining the relationship between agile EA and GDAD communication ("how"), and (2) this research extends the previous theories in agility and communication (i.e. Common Ground theory [Clark & Brennan 1991]). However, due to the need to reach a more valid and practical theory (model), this research investigates some case studies. Hence, this research study posits itself to have an exploratory dimension as well as explanatory, since it is important to formulate more structured theory about using agile EA for GDAD communication.

5.3.2.2 Research Method Selection

Mingers (2001) argued that using one paradigm and only one method to collect the evidence is not applicable due to the three reasons: (1) the characteristics of any paradigm cannot be totally separated from other paradigms' characteristics since it is not fully recognised that research methods are completely internal to a single paradigm, (2) it is not fully possible, in practice, to separate the subject from the object, and (3) different paradigms provide different perspectives that are more complex than the set of theories bound to any paradigm, in real context.

The ontology and methodology of the research should be linked (Dobson 2001). Therefore, the most suitable methodology of the research can be established by identifying the objectives or purposes of the research. However, this is not absolute in all cases. For example, both positivist and interpretivist research can use case studies to collect the data. In one hand, positivist assumes that the researcher should be objective when collecting research evidence in order to test hypothetic-deductive theories (Walsham 1995). Quantitative approach enables studying the phenomenon objectively (without influencing it or being influenced by it). Therefore, quantitative approach can avoid bias through logical and mathematical calculation. Hence, quantitative approach can be assumed to be based on positivism (Sale et al. 2002). Quantitative approach has its origins in the natural sciences but now is widely used in the social sciences such as IS (Myers 1997). Survey is the most practised method of collecting data in the

quantitative approach (Myers 1997). Quantitative approach is considered more suitable when the research problem evolves from the existing body of literature where the constructs and variables are known, and the relevant theories can be tested (Creswell 2009).

On the other hand, interpretivist assumes that reality is a social construct but it is subjective and based on shared meanings (Myers 1997). Statistical tests may be not enough to understand a problem that exists in a social context (Babbie 2001). In the qualitative research, the researcher and the object of the study are interactively linked and the emphasis is on the shared meanings and on the process (Babbie 2001). Moreover, qualitative approach enhances the in-depth understanding of the phenomenon. Therefore, qualitative approach is based on interpretivism (Sale et al. 2002). Qualitative approach has its origins in the social sciences (Myers 1997). Qualitative research can be conducted under any research paradigm depending on the information to be obtained and the nature of the research (Creswell 2009). Qualitative approach is more suited to the exploratory study where little is known about research problem, where constructs and variables are unknown, where there is no relevant theory to be tested, and where the researcher seeks in-depth understanding of a phenomenon in a certain context (Creswell 2009).

Mingers (2003) lists the methods that can be used in positivist and interpretive research. For positivism, the methods are:

1. Survey (questionnaire or instrument): it consists of all forms of data production involving the circulation of a pre-structured set of questions,
2. case study (both exploratory and confirmatory): a particular research state is important in its own right, and cannot be abstracted from its context,
3. observation, measurement and statistical analysis: they involve internally or externally published data and direct observation, recording or measurement. The data produced are generally quantitative,
4. controlled experiment: it involves both laboratory and field experiments. The classification includes any statistical analysis of the results, and

5. simulation: it involves the artificial production of data in such a way that it is representative of some aspect of a relevant real situation.

For interpretive research, the methods are:

1. Interview: it is a real-time conversation between researcher and interviewee (candidate) that seeks the candidate's personal views,
2. grounded theory: developing theory from, or grounded in, empirical social research. The approach uses data from a range of sources in order to generate theories that plausibly explain relationships among the concepts within the data. It does not accept an independent, pre-existing reality about which truth can be discovered,
3. ethnography: the researchers engaging themselves in the language, practices, and values of a particular organisation. The aim is to be able to understand what happens through the eyes of the people involved,
4. participant observation: the researcher becomes an active participant in the situation, usually, but not always, without the knowledge of the other people involved, and
5. qualitative content analysis: analysis of texts for the occurrence of specific terms or categories. Qualitative content analysis stems its categories from the material itself in a more interpretive way, recognising the role of the analyst in doing this.

There is no single research method which is universally applicable; all have their strengths and weakness (Mingers 2001). Some researchers claimed that using mixed-method approach in the same study is impossible due to the incommensurability of the research paradigm, since each paradigm has its own fundamental assumptions of the process of object inquiry that differs from the other paradigms (Mingers 2001). Another view argues (e.g., critical realists) that the multifaceted nature of social, organisational and phenomenological reality cannot be investigated by a single method (Goles & Hirschheim 2000).

5.3.2.3 Research Methods for this Study

Two principles, in general, have been used to develop a systematic body of knowledge (Bernard & Bernard 2012):

1. Deduction: the deductive reasoning method is valid if a conclusion necessarily follows from the premises (prior theories, findings, statements, or conditions). Deductive methods tend to move from the general to the specific, so a deductive argument is sound if it is valid and its premises are true. An example of the deductive research method is using survey for an explanatory study (Bernard & Bernard 2012).

2. Induction: the inductive reasoning method begins with small group of observations that include; evaluates similar events or subjects, detects patterns and regularities, and formulates hypotheses to develop general conclusions or theories. An example of the inductive research method is using interview for an exploratory study (Bernard & Bernard 2012).

Inductive method is different from deductive method in terms of order. Induction begins with an observation which displays a pattern. A hypothesis is formed from the pattern, and is built into a theory (Bernard & Bernard 2012). Deduction occurs in the reverse order. It begins with a pre-existing theory which develops into a hypothesis. Observations are made to reject or confirm the hypothesis (Bernard & Bernard 2012). Qualitative research is usually associated with inductive reasoning, while quantitative research usually adopts deductive reasoning (Creswell 2009).

This study takes the positivist and the interpretivist perspectives. This study adopts mixed-method approach. Hence, both the quantitative and qualitative approaches are used to investigate the phenomenon. The reasons for adopting this approach is that quantitative method enables testing relationships between the research model constructs and provide evidence to support the research hypotheses by employing objective measurements to collect research evidence (Walsham 1995). Moreover, qualitative method helps in validating the findings of the quantitative method, provides deeper explanations and results, and reveals the complex dynamics of tensions among the

research model constructs (Gable 1994). It also allows for comparisons of the theoretical results with the actual practice (Venkatesh et al. 2013). As mentioned in Section 5.3.1.4, the main paradigm used in this study is the positivist perspective (survey method) and the interpretivist perspective (case study method) will be used to validate the positivist findings. Moreover, inductive reasoning will be secondarily used in this study in addition to deductive reasoning as the main method.

5.3.3 Theoretical Foundation

IS research is characterised by a theoretical foundation and is expected to provide an explanation to the employed theories in the research (Gregor 2006; Gregor & Klien 2014). Since the beginning of IS revolution, academics have called for a solid theoretical foundation research (Gregor 2006). However, what composes theory and what it represented is still arguable discussion (Gregor 2006). Theory can be defined as the description that intends to explain or predict a phenomenon (Weber 2003). Gregor (2006) define the theory as a body of knowledge. Depending on the purpose of theory, Gregor (2006) classified theories into five types (characteristics):

1. Analysis: answer the question "What is". The theory does not extend beyond analysis and description. No causal relationships among phenomena are specified and no predictions are made,
2. explanation: answer the questions "What is", "How", "Why", "When" and "Where". The theory provides explanations but does not aim to predict with any precision. There are no testable propositions,
3. prediction: answer the questions "What is" and "What will be". The theory provides predictions and has testable propositions but does not have well-developed justificatory causal explanations,
4. explanation and Prediction: answer the questions "What is", "How", "Why", "When", "Where" and "What will be". It provides predictions and has both testable proposition and causal explanations, and
5. design and action: answer the question "how to do something". The theory gives explicit prescriptions (e.g. methods, techniques, principles of form of function) for constructing and artefacts.

As mentioned previously, the foundation theory of this study is the Common Ground theory (see Section 4.2). The research model of this study can be seen to possess some of each of the above characteristics. Firstly, it has the ability of analysis, which describes and classifies dimensions of an element of interest. The relevance to this study can be seen through the deriving of the prior model depending on the relevant literature and the survey. Secondly, with regards to explanation characteristic, the research model is relevant to this characteristic in its ability to explain the re-specifications of the model according to the survey and the case studies analysis. Thirdly, with regards to prediction characteristic, the research model is relevant to this characteristic in its ability to predict results from a set of explanatory factors without explaining the causal relationship between the dependent and independent variables. For example, regression analyses are employed in the survey data analysis to discuss the interrelationships between agile EA and communication, and with project performance. In other words, the regression analyses are used to assist in predicting the dependent variable(s) of interest using a set of predefined independent variables. Finally, with regards to design and action characteristics, the research model is relevant to this characteristic since it was carefully derived from the previous theories in communication, EA, and agility. The survey and case study questions are designed with careful attention to guidelines and procedures recommended in the literature.

However, this study can be described as being of an "Explanation and Prediction" type, since the research model is relevant to these characteristics in its ability to predict the behaviour of the dependent variable (e.g., communication) depending on the independent variable (e.g., agile EA). Moreover, this model is able to causally explain the relation between the independent and dependent variables.

This study follows a well-structured approach that has been provided by Berthon et al. (2002) when extending the Common Ground theory. In addition, this study follows a well formulated theory building and testing approach to develop a standard instrument, following the recommendations of Lewis et al. (2005). Moreover, this study follows the Gable (1994) and Dube and Pare (2003) guidelines of conducting a mixed-method research (i.e. using survey and case study methods).

5.3.4 Research Strategy

In this research, there is a clear need to combine both the quantitative and qualitative approaches to satisfy all the objectives of the research (Creswell 2009). One way to combine both approaches is by combining both approaches methods (Creswell 2009). To combine the two approaches methods, the two features of the research situations should be combined; multidimensionality and the activities included at different stages of the research (Mingers 2001). This combination results in different designs (strategies) of mixed-method research such as (Mingers 2001):

1. Multi-methodology: different methods and different paradigms combined and used to implement a task,
2. multi-level: conducting the research at different levels inside an organisation using different methods,
3. sequential: using different methods in sequence such that each method feeds the next method,
4. parallel: using different methods in same time and the results feeding into each other, and
5. dominant: one method used as the main method and other methods used to help implementing the task.

Amongst these strategies, the sequential strategy is believed to be the most suited design for this study. The sequential strategy is identified by the use of a single data collection method at a time (Mingers 2001). The findings of the first phase are fed to the second phase, as shown in Figure 5.1. This strategy provides this research with the advantages of both quantitative and qualitative approaches to address all objectives of the research (Creswell 2009).

In this research, quantitative approach (e.g., surveys with large sample sizes and generalisability) is used as the primary approach to collect the data. The findings of the quantitative methods feed the secondary phase (data collection) qualitative approach

(e.g., in-depth details through semi-structured interviews). In this research the findings of quantitative methods are validated by using qualitative methods.

5.4 Research Data Collection Methods

The research topic itself and the type of research questions are the fundamental considerations in choosing a research method (Yin 2009). Moreover, factors such as time, resource constraints, and access issues are essential to be considered (Yin 2009). Based on the research topic and research questions this study aims to investigate the relationships between agile EA, communication, and project performance in GDAD environment. This is done by validating a set of propositions between these three constructs. Therefore, a research method was needed to facilitate explanatory and confirmatory investigation.

The focus of this research is not on measuring the effect of variables via an experimental setting, but rather on investigating the inter-relationships between variables (i.e., agile EA, communication) in achieving high project performance. Thus, an experimental design is considered not ideal for the current research objective. Experiment suited the situation where the investigator has control over the sample through direct, precise, and systematic manipulation of the phenomenon under investigation (Yin 2009). By contrast, this study seeks unconstrained dynamics of agile projects in GDAD context. Similarly, action research was not considered for this study as it would involve the researcher in the way of the real project real actions. This is opposed to the objectives of the study, which seeks the real actions and practices GDAD agilest.

Ethnography could have been used to address the objectives of this study. However, ethnographical research design is based almost entirely on fieldwork that requires the complete engagement of the researcher in the GDAD communication phenomenon for a long period of time. Such a method was not possible for this study as the scope of ethnography is far more expensive than was required for this study. Moreover, the limited existing research on agile EA and agile communication in GDAD environment

reduces the aptness of archival and historical research designs, which require a noticeably larger number of previous works in the similar research domain.

Table 5.2: The settings of this research endeavour

Research Approach	Research Approach Objectives	Data Collection Method	Data Collection Time	Data Analysis
Qualitative	• Identify the research model constructs and the relationships among them • Identify and define the dimensions and aspects of agile EA, communication, and software performance	• Semi-structured interviews (Face-to-face, email)	January 5, 2015 – January 25, 2015	A non-automated content technique
Quantitative	1. Prepare and validate the instrument measures 2. Examine the relationships between: agile EA and GDAD active communication, agile EA and GDAD performance, and GDAD active communication and GDAD performance	1. Pilot survey (Email-based) 2. Full-Scale survey (Online-SurveyMonkey, email)	1. February 10, 2015 – March 10, 2015 2. May 1. 2015 -October 19, 2015	1. SPSS v. 16 package 2. SPSS v. 16 and SmartPLS v. 3 software
Qualitative	• Understand the relationships between agile EA, GDAD active communication, and GDAD performance • Identify other dimensions or aspects for GDAD active communication and performance	• Semi-structured interviews (Face-to-face, phone) • Comments parts in the survey	March 20, 2016 – May 10, 2016	• A non-automated content analysis technique

In this study, the constructs have been identified from the previous literature and then qualitatively evaluated (see Section 5.5). Moreover, related theories are used to conceptualise these constructs. Therefore, the most appropriate research method for this study is a survey, as it helps to quantify the measurements and test the hypotheses. In addition, survey provides high generalisability of the findings, which is one of the main aims of the study. Also, the survey method has the ability to remotely conduct the survey, which is the case of this study where most of the participants were not from Australia and were contacted via emails or phone. Furthermore, to validate the survey findings and obtaining in-depth details and explanations, case study method is used as a

secondary data collection method. Sections 5.6 and 5.7 discuss both survey and case study methods in detail. Table 5.2 provides a description of this study setting that includes the research approaches, approach objectives, data collection method and techniques, data collecting times, and data analysis techniques.

5.5 Preliminary Research Model Evaluation[4]

All constructs of the research model should be conceptualised and operationalised before collecting the data (Cavana et al. 2001; Neuman 2003; Zikmund et al. 2012). The measurement process starts with conceptualisation, and then operationalisation of each construct. Conceptualisation is the process of refining the construct by giving it a conceptual or theoretical definition (Neuman 2003). Operationalisation is the process that describes how to measure the conceptualised construct (Straub et al. 2005). This research starts with an abstract model (see Section 4.5). The related constructs and relationships between them are detected based on the literature to develop a conceptual theoretical model (the research model). This model consists of hypotheses with propositions and measurement of variables. The first stage of testing the model and the hypotheses was by evaluating this model. The hypotheses are then empirically tested through survey and interviews, and the findings are generalised providing a broader statement.

For extra conceptualisation of the constructs, the initial (preliminary) evaluation of the proposed model was conducted by involving five experts from both academia and industry in agile development. Three of them were from GDAD industry; a Scrum Master, a developer and an architect. Two of them worked as agile developers and now are assistant professors teaching agile development and agile enterprise architecture subjects. Two experts were asked the questions during 60-minute semi-structured face-to-face interviews, and three experts were emailed the model and questions (Gable 1994). The asked questions included:

- Is the design of the model clear, well thought out and easy to understand?

4 This section appeared in Alzoubi & Gill 2016

- Does it provide the necessary (relevant and important) constructs?
- Does it provide the necessary (relevant and important) relationships between the constructs?
- Does it provide the necessary (relevant and important) hypothesis?
- Is it suitable for its intended purpose?

The feedback supports the model design and its understandability, its constructs and relationships between different variables, and its suitability for the purpose of research. One expert wrote:

> *I think the model has been rigorously built and the relationships between different variables have been clearly identified*.

The feedback supports the role of agile EA and the role of the two communication dimensions; efficiency and effectiveness in GDAD. One expert mentioned:

> *Investigating agile EA role in the distributed agile environment seems to be very interesting and has a lot of potential*.

Another expert mentioned:

> *When we talk about communication, we are assuming to be quick and with focused message*.

We expected some disagreement on the definitions of functionality and quality variables from the interviews. Some experts refer to functionality as a part of quality. One expert mentioned:

> *Functionality is a part of quality since without achieving its functionality, software cannot be assumed of high quality*.

However, it is envisioned that functionality and quality are different concepts (Chow & Cao 2008; Dyba et al. 2007; Misra et al. 2009) so we included them in the model as

separate variables. Moreover, a direct relationship between agile EA and GDAD performance was included in the model since some feedback assumed that there is direct effect of agile EA on project performance. One expert suggested that:

> *I believe EA have more effect on project performance than on communication*.

Considering all feedback, the updated model was sent via email to the same above expert group for evaluation. Based on the second feedback, we preliminary validated the research model (Figure 4.5) for further research.

5.6 Survey

The primary data collection method used for this study is survey method. Survey method is considered to be the most widely used method in the IS research (Pinsonneault & Kraemer 1993). Survey is broadly described as encompassing any measurement procedures that involve asking questions of participants (Pinsonneault & Kraemer 1993). The survey method refers to quantitative analysis for a big data from a large number of participants and in new research areas where little theory has been developed (Gable 1994; Pinsonneault & Kraemer 1993). Survey method is recommended for investigating the relationships between a range of constructs, variables, or factors across a large population and is recommended for verification and validation purposes (Gable 1994). Survey is an appropriate method to answer research questions ‘what’, ‘who’, ‘where’, ‘how many’ and ‘how much’ (Yin 2009). Moreover, survey uses structured and predefined questions to collect information (questionnaire), and it aims to generalise the findings to the whole population of the study (Pinsonneault & Kraemer 1993).

The survey instrument can take many forms such as a written document, an online questionnaire, a face-to-face interview, or a telephone interview that should be completed by the participant (i.e. the person being surveyed) [Pinsonneault & Kraemer 1993]. Gable (1994) summarised some common features for the survey: medium controllability, medium repeatability, medium deductivity, medium discoverability,

medium representability, and high generalisability. Moreover, Gable (1994) listed the strengths and weakness of survey method as follows:

Strengths:

- Survey is useful in describing the characteristics of a large population,
- survey is relatively inexpensive,
- survey can be administered from remote locations using mail, email or telephone,
- many questions can be asked about a given topic giving considerable flexibility to the analysis and how the questions will be administered: as face-to-face interviews, by telephone, as group administered written or oral survey, or by electronic means, and
- standardised questions make measurement more precise by enforcing uniform definitions upon the participants, and ensures that similar data can be collected from groups then interpreted comparatively (between group study).

Weaknesses:

- The survey method that relies on standardisation forces the researcher to develop general questions,
- the researcher must ensure that a large number of the selected sample will reply,
- it may be hard for participants to recall information or to tell the truth about a controversial question,
- survey method is inflexible in that it requires the initial study design (the tool and administration of the tool) to remain unchanged throughout the data collection, and
- Survey method, in general, can seldom deal with the study context (excluding some interview approaches).

Pinsonneault and Kraemer (1993) classify survey research in terms of its purpose into three categories:

1. Explanatory survey: the purpose of explanatory surveys is to test a theory and the causal relations within that theory,
2. exploratory survey: the primary use of this survey is to become more familiar with a subject and to examine any initial concepts, and
3. descriptive survey: the primary goal of this survey is to examine what events, situations, attitudes or opinions are happening in a population.

Pinsonneault and Kraemer (1993) further classify surveys by their design, into two general types:

1. Cross-sectional surveys: this survey examines the sample population during a single time interval, with only one point in time for data collection, and
2. longitudinal surveys: this survey specifically examines the variables within the study over time.

The survey research method design includes a number of steps that are covered in the following sections. Section 5.6.1 discusses the instrument development, Section 5.6.2 discusses the instrument validation, the consent form and information sheet used in the survey are discussed in Section 5.6.3, the questionnaire is described in Section 5.6.4, sample design is discussed in Section 5.6.5, the comments parts used in the questionnaire is discussed in Section 5.6.6, data collection method used and data analysis approach are discussed in Section 5.6.7 and Section 5.6.8, respectively.

5.6.1 Research Instrument Development

This study follows a well formulated theory building and testing approach to develop a standard instrument, following the recommendations of Lewis et al. (2005). The initial pool of items (75 items) was distilled from the previous empirical studies (e.g., Herbsleb & Mockus 2003; Lee & Xia 2010; Mahaney & Lederer 2006) and from the definition of each construct. Theoretical constructs were operationalised using validated items from prior research and the experts' feedback. The initial pool of items is shown in Appendix5.1a. Then, the initial research instrument (50 items) was emailed to a group

of five experts (see Appendix5.1b) from both academia and industry in the field of ASD (see Section 5.5). The output of the expert feedback was 40 items, as shown in Table 5.3.

The experts helped in evaluating the fit between each item and associated construct. Based on the feedback, we dropped 17 items and 33 were reworded and new 7 items were created for agile EA variable where more focus was given to the role of solution architecture. One expert wrote:

> *In an ideal organisation, EA is used to produce a solution architecture which will be used to guide agile teams.*

The items of communication efficiency and effectiveness were refined to focus on communication and GDAD performance enhancement. One expert suggested that:

> *The questions should focus on asking how communication is going to be enhanced using "agile EA driven GDAD communication" rather than asking about the efficiency and effectiveness inside an organisation.*

The way of how agile EA and GDAD communication are related was reconsidered such that new questions were included in the survey questionnaire. One architect suggested that:

> *I think the definition of the EA should be clarified and then the link between the two (EA and GDAD communication) should be clarified to specify how EA is going to address the stated problems.*

Table 5.3: The set of questionnaire after experts and pilot feedback

Expert Feedback	Pilot Test Feedback
Agile EA (AEA)	
1. Enterprise architecture framework and the framework of distributed agile development are aligned 2. Enterprise architecture documentations are regularly updated to align with the projects in distributed agile development 3. Enterprise architecture is used to define projects/programs (e.g., business/IT gap analysis) in distributed agile development 4. Enterprise architecture is used to assess major project investment in distributed agile development 5. Solution architecture, as a part of enterprise architecture, guides the projects at program levels and project levels in distributed agile development 6. Solution architecture evolves from small iterations, and the changes in solution architecture are reflected in enterprise architecture 7. Enterprise architecture is used to govern project implementation in distributed agile development 8. I have enough information about my organisation plans and strategy to do my job 9. I have enough information about other teams included in my project to do my job 10. Results from meetings organised by colleagues from other sites are always visible to me	1. Enterprise architecture framework and the framework of distributed agile development are aligned. 2. Enterprise architecture documentations are regularly updated to align with the projects in distributed agile development 3. Enterprise architecture is used to define projects/programs (e.g., business/IT gap analysis) in distributed agile development 4. Enterprise architecture is used to assess major project investment in distributed agile development 5. Solution architecture, as a part of enterprise architecture, guides the projects at program levels and project levels in distributed agile development 6. Solution architecture evolves from small iterations, and the changes in solution architecture are reflected in enterprise architecture 7. Enterprise architecture is used to govern project implementation in distributed agile development 8. I have enough information about my organisation plans and strategy to do my job 9. I have enough information about other teams included in my project to do my job
Communication Efficiency (EFFIC)	
1. I lose time trying to figure out who to contact regarding my work 2. People I need to communicate with are difficult to find 3. I often get timely information about changes in current plans 4. Our organisation encourages rapid communication 5. I often do not get timely information due to high cost of communication 6. The people needed to communicate with are reached quickly 7. The people needed to communicate with are reached easily	1. Information needed about distributed agile development project is achieved quickly 2. Information needed about distributed agile development project is achieved easily 3. The stakeholders needed to communicate with in distributed agile development are reached quickly 4. The stakeholders needed to communicate with in distributed agile development are reached easily 5. The cost of communication (e.g., less travelling to meet face-to-face) in distributed agile development is decreased using EA 6. Our organisation encourages rapid communication
Communication Effectiveness (EFFECT)	

1. I often get useful work-related information through casual conversations 2. When work is assigned, everyone is clear about his or her task 3. Most of the information I receive on a daily basis is accurate 4. Most of the information I receive on a daily basis is reliable 5. Most of the information I receive on a daily basis is useful 6. I participate actively in the online discussions with other team members	1. All distributed agile development team members are clear about their tasks 2. Enough information is provided about customer requirements and project progress to distributed agile development team members 3. Detailed information is provided from distributed stakeholders 4. Accurate information is provided from distributed stakeholders 5. Most of the information I receive on a daily basis is useful
On-Time Completion (TIME)	
1. The project was completed late according to the original schedule 2. Project start date: _____, planned completion date: _____, actual completion date _____ 3. The team members complete and on-time submission of final project deliverables	1. Distributed agile development projects is completed on-time according to the original schedule 2. Distributed agile development tams complete their tasks on-time according to the original schedule 3. Project start date: _____, planned completion date: _____, actual completion date _____
On-Budget Completion (BUDGT)	
1. The team members complete their tasks according to the original budget 2. The project was completed over budget according to the original budget 3. Planned project cost: (in dollar) _____, actual project cost: (in dollar)	1. Distributed agile development projects is completed on-budget according to the original budget 2. Distributed agile development tams complete their tasks on-budget according to the original budget 3. Planned project cost: (in dollar) _____, actual project cost: (in dollar) ____
Functionality (FUNC)	
1. The software delivered by the project achieved its functional goals 2. The software delivered by the project met end-user requirements 3. The capabilities of the software fit end-user needs 4. The software met technical requirements 5. The project that has been developed works 6. This project has directly benefited the intended users either through increasing efficiency or employee effectiveness 7. Important clients, directly affected by this project, will make use of it	1. Distributed agile development project achieves its functional goals 2. Distributed agile development project meets its technical functional requirements 3. Distributed agile development project meets customer's functional requirements 4. The project that has been developed works 5. Important clients, directly affected by this project, will make use of it
Quality	

1. Given the problem for which it was developed, this project seems to do the best job of solving that problem, i.e., it was the best choice among the set of alternatives 2. Use of this project has directly led to improved or more effective decision making or performance for the clients 3. This project will have a positive impact on those who make use of it 4. The results of this project represent a definite improvement over the way the clients used to perform these activities	1. Distributed agile development project solves the given problem 2. Distributed agile development project improves the way of customers' use to perform their activities 3. Distributed agile development project achieves customer's satisfaction 4. Distributed agile development project represents a definite improvement over the way the clients used to perform these activities

Secondly, using snowball sampling technique which is recommended for exploratory research (Gregor & Klein 2014), pilot test was conducted by sending the instrument to five respondents (Product Owner, Scrum Master, an architect, two developers) based on the pre-established unit of analysis. The unit of analysis for this study is an individual who works in GDAD environment. The respondents were asked to complete the instrument, provide comments on difficulties in completing the instrument, and offer suggestions for improvement, including specifying any additional item statements they felt were missing or items that should be deleted. 5 items were dropped based on the feedback, and most of the items were reworded. The output of this step was adjusting and decreasing the number of items and questions to 35 items, as shown in Table 5.3.

Finally, the updated instrument was sent to a group of experts (same group as in Section 5.5) for item screening process. The purpose of this step is to empirically screen the items of the instrument applying a quantitative procedure developed by Lawshe (1975). Lawshe procedure determines whether each item on the instrument adequately represents the content domain of the construct. The experts were asked to evaluate the relevance of each item to its related construct on a three-point scale: '1=Not Relevant,' '2=Important (but not Essential),' '3=Essential' (Lewis et al. 2005). From the data provided by the expert panel, a content validity ratio (CVR) was computed for each item according to the following formula:

$$CVR= (n - N/2)/ (N/2)$$

Where N is the total number of respondents and n is the frequency count of the number of panellists rating the item as appropriate, either '3=Essential' or '2=Important (but Not Essential)' and '1=Not relevant'. Lawshe (1975) only utilised the '3=Essential' response category in computing the CVR. However, a less stringent criterion (both 2

and 3) could also be justified since responses of both ‘important (but not essential)’ and ‘essential’ are positive indicators of an item’s relevance to the construct (Lewis et al. 2005). In this study, we included both '3= essential' and '2= important' as the instrument was still in initial steps of evaluating process.

To filter the valid variables, a mean threshold of 0.5 was set. Therefore, those variables that have a mean between 0.5 and 1.0 and CVR value above 0.5 were accepted for the desired level of content validity for each item, whereas the variables with a mean and CVR values less than 0.5 were rejected and dropped from the instrument as an unacceptable level of content validity. Table 5.4 summarises the responses from the panel of experts and the quantitative analysis of the data using the mean and CVR value for each variable. The final number of items was 26 items. The up-dated final instrument is shown in Table 5.5.

Table 5.4: Items' results of the panel of experts' judgements

Item	n=1	n=2	n=3	Total accepted	Mean	CVR	Action
Agile EA1	1	2	2	4	0.80	0.60	Accepted
Agile EA2	0	2	3	5	1.00	1.00	Accepted
Agile EA3	0	1	4	5	1.00	1.00	Accepted
Agile EA4	1	2	2	4	1.00	0.60	Accepted
Agile EA5	0	0	5	5	1.00	1.00	Accepted
Agile EA6	0	1	4	5	1.00	1.00	Accepted
Agile EA7	1	2	2	4	.80	0.60	Accepted
Agile EA8	5	0	0	0	0.00	0.00	Rejected
Agile EA9	4	1	0	1	0.20	-0.60	Rejected
Communication Efficiency1	0	1	4	5	1.00	1.00	Accepted
Communication Efficiency2	0	1	4	5	1.00	1.00	Accepted
Communication Efficiency3	0	2	3	5	1.00	1.00	Accepted
Communication Efficiency4	0	2	3	5	1.00	1.00	Accepted
Communication Efficiency5	1	2	2	4	0.80	0.80	Accepted
Communication Efficiency6	3	2	0	2	0.40	-0.20	Rejected
Communication Effectiveness1	0	1	4	5	1.00	1.00	Accepted
Communication Effectiveness2	1	1	3	4	0.80	0.60	Accepted
Communication Effectiveness3	1	2	2	4	0.80	0.60	Accepted
Communication Effectiveness4	1	2	2	4	0.80	0.60	Accepted
Communication Effectiveness5	3	1	1	2	0.40	0.20	Rejected
On-Time Completion1	0	1	4	5	1.00	1.00	Accepted
On-Time Completion2	0	2	3	5	1.00	1.00	Accepted
On-Time Completion3	3	1	1	2	0.40	0.20	Rejected
On-Budget Completion1	0	1	4	5	1.00	1.00	Accepted
On-Budget Completion2	0	2	3	5	1.00	1.00	Accepted
On-Budget Completion3	3	1	1	2	0.40	0.20	Rejected
Functionality1	0	0	5	5	1.00	1.00	Accepted
Functionality2	1	1	3	4	0.80	0.60	Accepted
Functionality3	1	1	3	4	0.80	0.60	Accepted
Functionality4	3	2	0	2	0.40	0.20	Rejected
Functionality5	4	1	0	1	0.20	-0.60	Rejected
Quality1	1	1	3	4	0.80	0.60	Accepted
Quality2	0	1	4	5	1.00	1.00	Accepted
Quality3	0	2	3	5	1.00	1.00	Accepted
Quality4	3	2	0	2	0.40	-0.20	Rejected

Table 5.5: The research constructs and their items

Construct	Source of Items	Item
Agile Enterprise Architecture	New items built based on the experts' feedback	1. Enterprise architecture framework and the framework of GDAD are aligned 2. Enterprise architecture documentations are regularly updated to align with the projects in GDAD 3. Enterprise architecture is used to define projects/programs (e.g., business/IT gap analysis) in GDAD 4. Enterprise architecture is used to assess major project investment in GDAD 5. Solution architecture, as a part of enterprise architecture, guides the projects at program levels and project levels in GDAD 6. Solution architecture evolves from small iterations, and the changes in solution architecture are reflected in enterprise architecture 7. Enterprise architecture is used to govern project implementation in GDAD
Communication Efficiency	Adopted from: Herbsleb & Mockus 2003; Piccoli et al. 2004	1. Information needed about GDAD project is achieved quickly 2. Information needed about GDAD project is achieved easily 3. The stakeholders needed to communicate with in GDAD are reached quickly 4. The stakeholders needed to communicate with in GDAD are reached easily 5. The cost of communication (e.g., less travelling to meet face-to-face) in GDAD is decreased
Communication Effectiveness	Adopted from: Herbsleb & Mockus 2003; Piccoli et al. 2004	1. All GDAD team members are clear about their tasks 2. Enough information is provided about customer requirements and project progress to GDAD team members 3. Detailed information is provided from distributed stakeholders 4. Accurate information is provided from distributed stakeholders
On-Time Completion	Adopted from: Lee & Xia 2010; Yadav et al. 2009	1. GDAD project is completed on-time according to the original schedule 2. GDAD tams complete their tasks on-time according to the original schedule
On-Budget Completion	Adopted from: Jiang & Klein 2000; Lee & Xia 2010	1. GDAD project is completed on-budget according to the original budget 2. GDAD tams complete their tasks on-budget according to the original budget
Software Functionality	Adopted from: Lee & Xia 2010	1. GDAD project achieves its functional goals 2. GDAD project meets its technical functional requirements 3. GDAD project meets customer's functional requirements

Software Quality	Adopted from: Mahaney & Lederer 2006	1. GDAD project solves the given problem 2. GDAD project improves the way of customers' use to perform their activities 3. GDAD project achieves customer's satisfaction

5.6.2 Research Instrument Validation

Validating the research instrument is considered critical before conducting the full scale confirmatory empirical research (Straub 1989). The main instrument validity is done to ensure that a group of measurement items appropriately represent the concept (i.e. construct) under investigation (Straub et al. 2004). The instrument was validated through testing content validity and a pilot study.

5.6.2.1 Content Validity

Content validity refers to the issue of representation. That is, whether or not the instrumentation operates in a representative manner from all of the possible ways that could be used to measure the content of a given construct (Lewis et al. 2005; Straub et al. 2004). Some of the practical methods to measure content validity were identified through literature reviews and expert opinion (Straub et al. 2004). According to Lewis et al. (2005) guidelines, the measurement content validity can be achieved by pre-testing the instrument.

The instrument was sent to an academic expert on measurements who assessed instrumentation quality and the statements of items to ensure they reflected the intended sample frame. Moreover, the instrument was sent to three PhD scholars affiliated with the university who were asked to complete and critique the instrument. This is important for initial instrument design, such as format, content, understandability, terminology, and ease and speed of completion (Nunnally 1978). The wordings of the instrument items were fine-tuned, and the format of the questionnaire was improved as an output of this test. This step also ensured the face validity of the measurement, which according to Sekaran (2006) is the primary index of the content validity. Face validity evaluate if the instrument measures the constructs which it is supposed to measures (Sekaran 2006).

Two of the three academic affiliations were observed while completing the survey. This was done to calculate the approximate time needed to complete the survey and observed any difficulties in completing it. Their time average was about 17 minutes. This average was used as the base for the estimated time each participant needs to complete the survey (15-20 minutes was stipulated as the needed time).

5.6.2.2 Pilot Study

The pilot study is defined as the pre-testing study of a specific research instrument of investigation (Baker 1994). Performing a pilot study is important and helps in refining the research model, instrument, and hence increasing the accuracy of the research method and its results (Baker 1994). EFA was conducted in order to identify, reduce and validate the underlying factors of the construct (Tabachnick & Fidell 2007).

We modelled the indicators of communication efficiency, communication effectiveness, on-time completion, on-budget completion, software functionality, and software quality as reflective (caused by their latent constructs) measures (Petter et al. 2007). However, we modelled the indicators of agile EA as formative measures since its indicators are not expected to have covariation within the same latent construct and they are causes of, rather than caused by, their latent construct (Petter et al. 2007).

Assessment of measurement model includes the evaluation of reliability and convergent validity. Reflective and formative constructs are treated differently since for formative constructs, unlike reflective constructs, indicators are not expected to demonstrate internal consistency and correlations, unlike reflective constructs (Chin & Todd 1995). Instead, the relevance and level of contribution of each indicator was assessed by examining the absolute indicator weights. Reflective and formative evaluations are discussed in Section 5.6.2.2.2 and Section 5.6.2.2.3.

5.6.2.2.1 Pilot Study Sample Demographics

The unit of analysis for this study is an individual who works in GDAD environment and is using EA in his/ her development. A snowball sampling technique was utilised to identify the individuals who know other individuals that can provide us with rich

information. My supervisors provided some contacts of GDAD individuals. The criteria for the individual selection included that the individual should be a GDAD team member and has a willingness to participate. Although there is no common agreement about the sample size of the exploratory assessment, Hunt et al. (1982) recommended a sample size between 12 and 30 subjects.

The sample included 60 GDAD team members. They were contacted by email to complete the survey. A total of 45 surveys were returned, achieving 75 % survey response rate. 8 incomplete surveys were exempted from the analysis. Thus, 37 of the returned surveys were usable responses. Of the surveys analysed, as Table 5.6 shows, 20 respondents (54 %) were developers, 7 (19 %) were architects, 4 (10.8 %) were team leaders/Scrum Masters and 4 (10.8%) analysts, and 2 (5.4 %) QA/test. Most of the respondents had (2-4) years experience in GDAD.

Table 5.6: The demographic data of the pilot study

Variable	Value	Frequency	Percent %
Job Title	Developer	20	54
	Team leader/Scrum Master	4	10.8
	Analyst	4	10.8
	Architect	7	19
	QA/test	2	5.4
Industry	Banking and financial	20	54
	Automotive	5	13.5
	Telecommunications	8	21.7
	Other	4	10.8
GDAD Experience	< 2 years	5	13.5
	2-4 years	25	67.5
	5-10 years	7	19

5.6.2.2.2 Reflective Measurement Evaluation

Reflective measures were evaluated by measuring the instrument reliability and convergent validity (correlation between variables) using factor analysis. Firstly, we assessed the reliability of reflective constructs with Cronbach's alpha coefficient.

Reliability refers to the degree to which the measurement item is free of random errors and yields consistent and stable measures over time (Straub et al. 2004). Using reliability analysis, the researcher can determine the extent to which the items in each variable are related to each other (Straub et al. 2004). It also provides an overall index of internal consistency of the variables. Cronbach alpha is a direct function of both the number of items and their magnitude of inter-correlation, and is the lower bound to the test variance attributable to common variables among the items within each construct (Nunnally 1978). Using Cronbach alpha helps in excluding items from a variable or including them in another variable (sekaran 2006). High value of Cronbach's alpha indicates a high consistency in variance of the sample test scores (i.e. high reliability or accuracy of the statistical inferences from the data). Although some researchers determined 0.7 as a cut-off for the accepted reliability, Cronbach's alpha value greater than 0.6 is considered satisfactory in survey research (Hair et al 2010).

Table 5.7: Pilot study-reliability statistics of reflective constructs

Construct	Item	α if Item deleted	Cronbach's α	No. of Items
Efficiency	EFFIC1	0.910	0.935	5
	EFFIC2	0.918		
	EFFIC3	0.919		
	EFFIC4	0.929		
	EFFIC5	0.923		
Effectiveness	EFFECT1	0.851	0.886	4
	EFFECT2	0.841		
	EFFECT3	0.865		
	EFFECT4	0.856		
Time	TIME1	-	0.932	2
	TIME2	-		
Budget	BUDGT1	-	0.768	2
	BUDGT2	-		
Functionality	FUNC1	0.762	0.841	3
	FUNC2	0.842		
	FUNC3	0.726		
Quality	QLTY1	0.769	0.862	3
	QLTY2	0.800		
	QLTY3	0.846		

SPSS v.16 package was used to test the internal consistency for each construct's items individually. The results are presented in Table 5.7 (where; Efficiency =

communication efficiency, Effectiveness = communication effectiveness, Time = on-time completion, Budget = on-budget completion, Functionality = software functionality, Quality = software quality). All reflective constructs achieved scores above the recommended value of 0.70 for Cronbach's alpha, so we retained all the items (Hair et al. 2010).

Secondly, convergent validity was assessed using factor analysis. In running the exploratory factor analysis, principal component analysis, the VARIMAX rotation method, and a minimum eigenvalue of 1 were chosen as conditions for factor extraction. Items were allocated to a factor if their primary loading was greater than 0.5, if they did not cross-load onto more than one factor and if their communality is greater than 0.4 (Lewis et al. 2005). Moreover, Kaiser-Meyer-Olkin (KMO) value, which measures the sampling adequacy, should be greater than 0.5 (Hair et al. 2010). As shown in Table 5.8, six factors corresponding to the reflective constructs in our model were extracted. Kaiser-Meyer-Olkin (KMO) was found to be 0.78, which is well above the recommended value of 0.50. All item loadings on stipulated constructs were greater than 0.50, and all eigenvalues were greater than one as required.

Table 5.8: Pilot study-factor analysis of reflective constructs

Construct Items	Component					
	1	2	3	4	5	6
Communication efficiency						
EFFIC1	**0.854**	0.294	0.222	-0.120	-0.024	0.100
EFFIC2	**0.796**	0.355	0.301	-0.051	0.001	-0.093
EFFIC3	**0.791**	0.224	0.151	-0.448	-0.082	0.075
EFFIC4	**0.816**	0.167	-0.014	-0.212	0.070	-0.189
EFFIC5	**0.855**	0.143	0.068	-0.006	-0.078	-0.187
Communication effectiveness						
EFFECT1	-0.458	**0.698**	0.193	0.012	0.299	-0.177
EFFECT2	-0.355	**0.698**	0.056	-0.107	0.500	-0.090
EFFECT3	-0.195	**0.826**	-0.130	-0.278	0.023	0.267
EFFECT4	-0.466	**0.592**	0.255	-0.357	0.189	0.250
On-Time completion						
TIME1	0.138	0.095	0.120	0.357	**0.813**	0.192
TIME2	0.178	0.034	0.319	0.314	**0.818**	0.125
In-Budget completion						
BUDGT1	0.248	0.160	-0.363	0.402	-0.002	**0.668**
BUDGT2	-0.096	0.240	-0.185	0.047	-0.156	**0.819**
Functionality						
FUNC1	-0.235	-0.257	0.309	**0.712**	0.193	0.287
FUNC2	-0.362	-0.328	0.402	**0.625**	0.202	-0.184
FUNC3	-0.059	-0.325	0.475	**0.667**	0.197	-0.160
Quality						
QLTY1	-0.082	-0.123	**0.809**	-0.342	-0.178	-0.018
QLTY2	0.156	0.032	**0.773**	-0.366	0.172	-0.287
QLTY3	-0.204	0.176	**0.740**	-0.345	-0.258	-0.045
Eigen value	3.971	2.989	2.352	2.279	1.874	1.625
Variance Extracted	26.887	20.613	13.258	13.004	6.525	4.476
Cumulative Variance (%)	26.887	47.500	60.759	73.763	80.288	84.764
Unrotated Variance (%)	34.439	30.079	6.946	5.569	4.481	3.250

5.6.2.2.3 Formative Measure Evaluation

The reliability and convergent and discriminant validity for the formative construct AEA (agile EA) was assessed using the collinearity which refers to the existence of high correlations between formative indicators (Hair et al. 2014). Collinearity was assessed by computing tolerance and VIF (variance inflation factor= 1/tolerance) using IBM SPSS v. 16 Statistics. Tolerance represents the amount of variance of one formative indicator not explained by other indicators in the same construct. A tolerance value of 0.2 or lower and a VIF value of 5 or higher indicate a potential collinearity problem (Hair et al. 2014). If the VIF statistic is greater than 5, the conflicting item should be removed as long as the overall content validity of the construct measures is not compromised (Hair et al. 2014). The tolerance and VIF estimates for the measures of AEA are shown in Table 5.9. The results suggested that the reliability of AEA is supported and all indicators can be retained.

Table 5.9: Pilot study-collinearity test of formative construct

Item	AEA1	AEA2	AEA3	AEA4	AEA5	AEA6	AEA7
VIF	4.073	3.405	2.276	2.758	3.410	2.281	3.445
Tolerance	0.245	0.293	0.440	0.362	0.293	0.438	0.290

5.6.3 The Consent Form and Information Sheet

The consent form and the information sheet were presented at the beginning of the survey explaining the purpose of the study. The form emphasises the privacy of respondents and the confidentiality of collected information. The researcher and researcher's supervisor as well as university details were presented in the form also. Moreover, an indication about the ethics approval by Human Research Ethics Committee (HREC), at the University of Technology Sydney, is included in the form.

Following the consent form, an informative introduction was presented to give the respondents a general understanding about agile EA and its use in GDAD, and its impact on development performance. The consent form and the introduction can be found in Appendix 5.2a.

5.6.4 The Questionnaire of the Survey

At the beginning of the survey, the research approach was briefly introduced. The first section of the survey asked the participants to evaluate the use of agile EA in their organisation by rating a set of statements. In the next section, the participants were asked to evaluate the communication process (communication efficiency and communication effectiveness) when they communicate with other distributed team members, by rating a set of statements. Thirdly, the respondents were asked to evaluate their GDAD development performance (on-time completion, on-budget completion, functionality, and quality) by rating a set of statements. After each of the above sections, an empty space was left for other comments regarding that section. Finally, the participants were asked to answer a set of demographic questions. The aim of the demographic questions was to provide the study with a general descriptive profile about the sample population. The full questionnaire can be found in Appendix 5.2b.

All Items of the questionnaire were measured using a seven-point Likert scale. The scale ranges from "strongly disagree" to "strongly agree" or “very little” to “very much”. Likert scale was chosen since it is the most relevant scale of studies that seek to explore theories of attitudes or patterns (Oppenheim 2000). Moreover, Likert scales are the most frequently used scales in IS research (Sekaran 2006). Since the reliability of scale increases with increasing the increment of the number of choices (Lissitz & Green 1975), seven-point were used in this study. Moreover, Likert scale is a relatively simple response scale, which does not add substantial amounts of construct-irrelevant variance as for complex scales (Morgeson & Humphrey 2006). Likert scale is an interval rating scale with responses which vary from strong refutation to strong affirmation, and mid-point representing the neutral response. Each response is considered equal in the scale regarding value loading or attitude (Sekaran 2006).

5.6.5 Sample Population

Sampling is the process of selecting a sample from a population for the study interest (Yin 2009). After studying the sample, the generalisation can be made to the whole population (Yin 2009). The selection criteria which was used for this study includes

experience in GDAD, using EA elements in the development, and a minimum of two distributed teams or two distributed team members. The sample for this study was 500 agilest chosen from different organisations that employ agile EA driven GDAD. The population is all software developers and software organisations that use GDAD.

The background questions were used to profile the respondents and to summarise the relevant information regarding their organisations. Testing demographic measures is not to establish a causal relationship with the study variables; rather it is just to provide a background data to establish a common perspective on the respondents and organisations (Yin 2009). Some demographics were asked such as the industry of the organisation, development approach of the organisation, participants' experience, and number of teams included. For more information, see Section 6.2.

It has been highly censured that researchers are less concerned about the adequacy of the sample size (Marcoulides et al. 2009). Researchers should obtain adequate size of their studies sample to ensure the stability of the estimation of prediction accuracy and to allow for deletion of low value coefficients in the model (Marcoulides & Saunders 2006). To ensure that this study has obtained an adequate sample size to conduct the quantitative analysis, 80 percent or greater level of power should be achieved by the research model, according to Cohen (1988). Alpha level of 0.05 and power level of 80 percent have been widely accepted in IS literature as the sufficient levels of confidence and power that enable researchers to reject the hypothesis when the factor correlation in the population is zero (Marcoulides & Saunders 2006). With these levels of confidence and power, it is reasonable to detect, at least, a medium correlation effect size *r* value equal to 0.3 (Cohen 1988, Straub et al. 2004). According to Hair et al. (2010), sample size of 200 or more should be enough to obtain the desired power level of 0.8. Moreover, according to Hair et al. (2014), the minimum sample size should be 10 times the maximum number of arrowheads pointing at a latent variable anywhere in the PLS model. Applied to this study, a sample size of at least 30 (10 * 3 = 30, maximum arrows number was 3) would be enough to run PLS analysis.

Nevertheless, for this study, a power analysis was conducted using G*power software (Faul et al. 2007). The one correlation test was conducted using the following inputs:

- Tails = two
- Effect size $r = 0.3$
- Significance level $\alpha = 0.05$
- Desired power $(1-\beta) = 0.80$
- Population correlation = 0

The result of the test indicates that a sample size of at least 84 was sufficient to do the quantitative analysis of this study. Therefore, the 160-sample size of this study was well above and adequate to carry out the PLS analysis.

5.6.6 The Comments Parts of the Survey

Although the survey was designed predominantly for quantitative analysis, it also included three sections for comments at the end of each main sections of the survey (i.e. agile EA, communication, and software performance) see Appendix 5.2b. The purpose of these comments was to solicit responses from the participants about the agile EA, communication, and performance of software developed. This technique is important to explore the problem in more details. The comments included broad personal opinions, concerns, real life experiences, and remarks that respondents were willing to share. All comments collected from the survey were analysed with the qualitative analysis (Ch.8).

5.6.7 Survey Data Collection Method

Survey can be communicated with potential participants and data can be collected using different methods such as face-to-face, phone, email, and web (Malhotra & Birks 2003). However, the selection of survey method should be justified upon several considerations such as the goal of the study, the participants' characteristics, and recourses constraints such as time and cost (Malhotra & Birks 2003).

After careful consideration, the web-based (SurveyMonkey tool) and email questionnaire techniques were used to communicate and collect the data of the survey. The nature of problem under investigation of this study makes use of the advantages of these techniques. The study seeks to gather the experiences and opinions of a large

number of users of GDAD. The survey questionnaire was sent to potential respondents in different industrial sectors (e.g., financial, telecommunications, and agencies) that are practicing agile software development in GDAD environment. Similarly, we did not restrict ourselves to particular organisation, country, or nationality. With this approach, we were able to survey GDAD professionals that are widely geographically distributed across continents. Using these techniques, the respondents were able to answer the questions in their free time, and answer them more thoughtfully by taking their own time. Moreover, from design perspective, the mail design for example or face-to-face techniques require substantial efforts to travel, collate, pack, and deliver the survey. In addition, substantial cost and time of collecting data are needed as a result of waiting for replies, which affect the speed of data collection (Malhotra & Birks 2003).

In this study, most responses were received through the web-based survey. The reason for this could be with the ease of completing the online questionnaire by following the survey link which was sent to respondents, rather than downloading the questionnaire and then emailing back the completed questionnaire to the researcher.

5.6.8 Quantitative Data Analysis Approach

This study used SPSS v.16 (IBM Corporation 2015) and SmartPLS M.3 (Ringle et al. 2015) statistical packages to test and analyse the data. Firstly, SPSS was used to test data, generate descriptive statistics, and generate summaries about data. The package was also used to obtain the independent t-Test and ANOVA-Test for different groups, Pearson's Chi-square Test to compare the correlation between different variables (Tabachnick & Fidell 2007). The independent t-Test (also called two sample t-test) is an inferential statistical test that aims in determining the differences in means of two groups (Tabachnick & Fidell 2007). Mainly, t-Test was used to examine the common method bias between the early and late data collected groups. Also, SPSS was used to generate Cronbach's alpha coefficients needed to examine the reliability of the measurements in both pilot and full studies. SPSS was used in the pilot study (see Section 5.6.2.2) and in the full-scale survey (see Ch.6) to obtain the EFA.

The use of appropriate modelling and statistical methods in analysing the quantitative data is a critical issue involved in testing the research model (Barclay et al. 1995). Hence, special considerations were taken to choose the technique that could handle some issues of the research model such as the complexity and formation agile EA as a formative construct. At the same time, the technique should provide all the required analyses that are usually reported in literature for similar studies. Therefore, and after taking all these considerations into account, the Structural Equation Modelling (SEM), and more precisely the PLS technique was chosen to test the components of the research model and its related hypotheses. Ch. 7 discusses, in details, PLS results and the reasons for choosing it as the most appropriate technique to test this study model and hypotheses.

5.7 Qualitative Case Study

The issue that emerges from the interaction of people, technology and process (organisation) may need to be qualitatively assessed for better understanding of the issue in order to develop theory or solve a problem (Gregor & Hevner 2013). As a form of qualitative research approach, case study represents an empirical investigation to a real-life phenomenon (case) that focuses on observation, reconstructions and analysis of that case especially when the boundaries between phenomenon and context are not clearly evident (Yin 2009). The use of case study method in the IS applications has increased during the last two decades (Pare 2004). Case study method has many applications such as investigation the individuals or group's knowledge related to a specific phenomenon, investigation the issues that result when the phenomenon shifts from technical to organisational level, gaining in-depth understanding of the complex cases and relationship between people, and exploring and testing a contextual phenomenon (Pare 2004). In addition, case study method can be used to generate a general conclusion from a limited number of cases (Yin 2009).

Case study method is used to investigate a predefined method without applying explicit manipulation to variables (Benbasat et al. 1987). Case study is the most preferred method when: (1) the research needs to answer the "How" or "Why" research questions,

(2) the researcher has little control over events, and (3) the focus of the research is on the current phenomenon within real-life context (Yin 2009).

Case study method includes either single or multiple cases (Yin 2009). Single case study method enables in-depth and close investigation to a phenomenon, and rich description and deep revealing to a phenomenon structure (Cavaye 1996). Multiple case studies are more suitable and desirable when the intent of the research is description, theory building, theory testing, or theory extension (Benbasat et al. 1987; Yin 2009). Multiple case studies are assumed more robust and can enhance the validity of the research findings (Yin 2009). However, case study method has been criticised by its inability to generalise the conclusions due to the small sample investigated and the lack of statistical reliability (Pare 2004).

Case study is used in this study as a secondary method to validate the findings of the quantitative approach (i.e. survey findings) since the combination of case study method attributes with other methods makes it applicable to many situations (Pare 2004). In other words, this means using both survey and case study methods to complement each other and overcome the shortage of each method. While survey helps in generalising the results, case study helps in gaining in-depth understanding of the relationships between agile EA, communication, and performance in the GDAD environment. The benefits from using the case studies are: (1) case studies findings will be used to validate and explain the supported hypotheses, (2) case studies findings provide explanations for the unsupported hypotheses, and (3) the results of case studies provide rich, and further insights on the complex relationships among the constructs and beyond the quantitative results (Gable 1994; Venkatesh et al. 2013).

Yin (2009) identifies five essential components for case studies: (1) study's question(s), (2) its propositions, (3) logic linking the data to the propositions, (4) its unit of analysis, and (5) the criteria for interpreting the findings. The questions asked during the interview can be found in Appendix 5.3b. The propositions (see Ch. 4) were derived from the literature. The logic linking the data from this study to the literature-based proposition is that using agile EA enhances the communication between team members and enhances software project performance in GDAD environment, and communication

efficiency and effectiveness enhance project performance in GDAD environment, which are consistent with the propositions in the literature. The unit of analysis is team members using agile EA in GDAD environment. Furthermore, in the process, the multiple case studies may qualify or find additional supporting evidence beyond that currently recognised in the literature. The primary criterion for interpreting the case-based data is an evidence of the relationships between agile EA, communication, and project performance in GDAD environment (the research model in Ch.4).

5.7.1 The Semi-Structured Approach of the Interviews

This research uses interview technique to collect data. Interviews may take one of the following forms (Jonker & Pennink 2010):

- Unstructured: this type of interviews is used and most appropriate when the interviewer has little knowledge about the research question and is trying to learn more about it.

- Semi-structured: A predefined set of questions is used. Additional questions might be asked during the interview as the researcher does not know exactly how the respondent will respond. The questions will be asked systematically and in consistent order; however, the researchers are allowed to digress, which means that researchers can probe beyond the respondent's answers. So, semi-structured interviews are an effective approach as they offer the needed information from respondents through predefined questions and offer the researcher additional information through going beyond the asked pre-questions.

- Structured: A predefined set of questions and limited responses to those questions are used. This approach can be more easily accomplished than the semi-structured approach; however, it offers less flexibility and less information.

This study uses semi-structured interviews as the secondary data collection method. Throughout the interview process, the researcher should follow the line of inquiry as

reflected in the case study protocol, and the questions should be asked in an unbiased manner (Yin 2009).

5.7.2 The Interview Protocol

Yin (2009) proposes two tactics, by documenting the procedures followed, to ensure reliability (minimise the errors and biases in a study): a case study protocol and a case study database. The case study protocol should contain procedures and general rules that should be followed in using the instruments, in addition to the interview instruments (Dube & Pare 2003). The case study database usually involves raw material (including interview transcripts, researcher's field notes, documents collected during data collection, and survey material), coded data, coding scheme, memos and other analytic material, and data displays (Dube & Pare 2003).

The interview protocol was developed, under the semi-structured approach, based on the study's objectives. The focus of the protocol was to guide the interview process. The protocol included the same concepts of the research model, such as agile EA, communication efficiency, and communication effectiveness; however, it also extended its reach to other areas where the quantitative approach was limited. The protocol, basically, included some procedures to be followed by the researcher in each interview. It also included extra sub-questions to expand the understanding of the question being discussed. The interview protocol can be found in Appendix 5.3b.

5.7.3 The Participants of the Interviews

The case study was designed to collect data from multiple GDAD projects that adopt agile EA-driven GDAD approach via a multiple-case design (Yin 2009), with each project representing a case. The sample for this study was 10 post hoc case studies of GDAD projects which were selected from the survey sample. Purposeful sampling was used to select cases that used GDAD approach to complete their projects. The choice of cases in this study was also opportunistic based on industry relationships initiated by the researcher. To preserve anonymity, the project cases are coded: A, B, up to J. The personal details of the participants were removed for anonymity reasons.

5.7.4 Interviews Process

All participants were informed about the interview by sending them emails and obtaining their agreement in conducting the interviews either face-to-face or on Skype. The email also included the study objectives and the questions which were going to be asked (in general) in order to give the participants enough time to prepare some information in advance for the interview. Most of interviews (10) were conducted face-to-face. The consent form (Appendix 5.3a) and the interview protocol (Appendix 5.3b) were followed in the interview. For the other Skype interviews (2), the participants were sent the consent form and returned it signed before conducting the interview.

While each interview was projected to last for 60 minutes, the actual interviews ranged between 15-70 minutes in length. Both voice recording technique and taking notes technique were used during each interview. The recording permission was asked at the beginning of each interview. At the beginning of each interview also, the researcher made sure he provided his contact information to the participants. A copy of the recorded interview and the transcript were sent back to the participants who asked for it to be sent to them, as per their preferred contacts. Participants were assured that they had the choice to modify the transcript or even withdraw it from any further stages of the research.

5.7.5 Qualitative Data Analysis Approach

Analysing and coding the qualitative data is one of the least developed and most difficult parts of conducting case studies (Eisenhardt 1989). Qualitative analysis can be defined as the process that identifies ideas, patterns of beliefs and significant themes from a set of data, and attempts to demonstrate support for them in order to transform them to meaningful findings (Aloudat 2010; Miles & Huberman 1994).

Content analysis approach was used to analyse the data from both interviews and comments parts in the survey (i.e. the unit of analysis). Content analysis is defined as the research method that uses a set of procedures to make valid inferences from text

(Weber 1990, p.9). Interactive model of analysis (Miles & Huberman 1994) was followed to guide the qualitative analysis. Initial categories were created based on the components of the research model (developed in Ch. 4). However, some additional information that emerged from the data was also included in the final category set.

In this study, cross-case analysis was done by searching for similarities and differences across the 10 cases to gain a deeper understanding of the phenomenon. We identified important comments from each case and compared them within and across cases. Key insights and shared patterns were achieved through several iterations of comparing and contrasting interview comments. The resulting findings were used to modify and extend the PLS findings.

5.8 Ethics Consideration

Ethics is an important component of IS research (Creswell 2009). Every researcher is obliged to define his or her actions and responsibilities at any time during the research life process by means of ethical principles (Creswell 2009). These principles represent the moral issues when conducting the research (Kimmel 1988). Ethical issues may turn out to be extremely complex and subtle and this leads to unresolvable situations, sometimes (Kimmel 1988). IS research involves collecting data from people, which raise the ethical issues. Therefore, the researcher should be able to keep the balance between the research methodology used and the respondent's rights and values that may be impacted by the research (Kimmel 1988). Therefore, the research principles should be applied and followed in a way that ensures both participant's ethical issues and research validity are not being impacted (Kimmel 1988). However, when research moves from general to more particular, these principles can be more problematic (Aloudat 2010). Therefore, the researcher may follow an established code of professional practice that defines responsibilities of the researcher and the boundaries of what is allowed and what is not (Kimmel 1988).

Hence, this study makes sure that all ethical requirements are fulfilled by following the University of Technology Sydney (UTS) research ethics. This means that the researcher must behave ethically with all research subjects, academics, practitioners and other

stakeholders. Compliant with University policy to protect human subjects, the survey and case study plans, including interview questions, were submitted to the UTS Engineering and Information Technology Faculty (FEIT)-Human Research Ethics Committee (HREC) panel for approval. The HREC panel formally approved the survey and case study plans (the approval is attached in Appendices 5.2a, 5.3a). The application includes the ethical considerations the researcher should adhere to. These embraced the confidentiality and privacy of the participant's identity and personal information, storage and access procedures to the collected data, the usage of the collected data, protection of any given responses from misrepresentation or exploitation, and the means through which the collected data will be disposed. In addition, comprehensive details about the aim of nature, duration, and the design of the study are also included in this application.

An indication of the ethics approval (consent form), which accompanied the researcher's details and the purpose of the study was sent with each survey or was added as the first part of the web-based version of the survey. It also, informed the participants that the participation in the survey is voluntary and they have the full right to withdraw from the survey at any stage they feel the need to or not to send the completed survey back to the researcher. In addition, the participants were informed that the survey is done anonymously as no personal identifiable information of any kind was required, and the information provided in the survey would be kept confidential and would not be accessed by anyone but the researcher himself.

Every interviewee was informed of all details as for the survey above. In addition, interviewees were assured that all their personal identifiable information would be totally removed prior to the subsequent analysis and only a pseudonym or a code would be used to identify the participant's area of expertise and the work experience period. Moreover, no interview was recorded without prior explicit approval from the participant.

5.9 Chapter Summary

This chapter discussed the research design of this study. It discussed the methods used to test the validity of the research model proposed in Ch.4. Primarily, the research follows a positivist paradigm and adopts quantitative approach using survey as the research method. Secondly, to validate the findings of the survey, the research follows an interpretivist paradigm and adopts a qualitative approach using interview as the research method.

The instrument design followed the rigorous procedure developed by Lewis, Templeton & Byrd (2005). This procedure begins with the definition of the domain constructs, generation of the initial instrument from the literature and evaluates it by experts, pre-testing, pilot testing, and item screening of the instrument to finalise the measurement items. Finally, the ethical considerations of this research were delineated in accordance with the conduct guidelines of ethics and professional conduct provided by the university, which define the responsibilities of the researcher at any given stage during the research life process.

Chapter 6 discusses the examination of the collected quantitative data. It discusses some statistical tests that are essential to obtain meaningful data and to prepare the data for the final analyses (i.e. SEM analyses). These tests include data screening, missing data analysis, outliers testing, testing for underlying assumptions (i.e. normality, linearity, homoscedasticity, and multicollinearity), EFA, and generalisability tests (i.e. common method bias and non-response bias).

Chapter 6 Survey Data Examination and Preparation

In Ch. 2 and Ch. 3 the research problem was distilled through scanning the previous literature gaps. In Ch. 4 the research model constructs and hypotheses were discussed. Ch. 5 discussed the research methodology. As we mentioned in Ch. 5, this research follows a sequential quantitative and qualitative approach to investigate and validate the research model and measurements. Quantitative method used the survey technique to collect data. This chapter discusses the first step in analysing the collected quantitative data; the process of preparing the data for the final analysis.

6.1 Introduction

This chapter discusses the process of screening (preparation and examination) of the collected data. Data screening, missing data analysis, and outliers' examination provide the data in a format and quality suitable (i.e. complied with statistical assumptions underlying multivariate techniques) for multivariate analysis. This process is essential for quantitative data for many reasons such as:

- organising the data correctly. This saves substantial time, prevents mistakes and minimises potential measurement error. This will result in less confusion and difficulty with the statistical analysis in the next step (Hair et al. 2010; Tabachnick & Fidell 2007),
- preparing the data. This helps in robustness with the requirements of modern and standardised data analysis tools such as SPSS, which means that data will be easily retrieved, transformed and maintained, and
- examining the data. This helps in verifying if the data satisfy the essential analysis requirements such as normality testing, internal consistency, generalisability, and bias issues (Hair et al. 2010).

We have followed systematic steps for data examination that are presented in the following sections and summarised at the end of this chapter. First, the sample demographics are discussed in Section 6.2. Second, in Section 6.3, Section 6.4 and Section 6.5, the data screening process (data were screened to detect any incomplete cases, inconsistencies missing values were analysed to delete irrelevant items and cases that have a high percentage of missing data beyond the threshold to be usable and to perform imputation on the remaining data, and outliers were identified and treated) and the survey data coding process are discussed. Third, Section 6.6 discusses the tests that were conducted to evaluate if the data complied with the assumptions of multivariate analysis. Fourth, Section 6.7 discusses the EFA which was conducted to test the validity and the unidimentionality of the scales. Fifth, the biases (common method and non-response) issues are discussed in Section 6.8, to ensure that the collected data can be generalised to the population. Finally, a brief summary of the chapter is presented in Section 6.9.

6.2 Demographic Analysis of the Sample

After testing the measurement model (questionnaire) and confirming its reliability through the pilot study, a convenient sample was chosen to distribute around 500 questionnaires. We targeted (sample) a number of agile EA-driven GDAD team members from different countries as the study population is all individuals or organisations who use GDAD. Since the focus of the study was in gathering the experiences of a large number of agile team members in GDAD environment, we sent the survey questionnaire to potential respondents in various industrial sectors (e.g., finance, telecommunications and healthcare). Similarly, we did not restrict ourselves to any particular organisation, nationality or project size due to the nature of this research. Additionally, the questionnaire was primarily collected through web-based (using SurveyMonkey tool) questionnaire and secondly by emails. The study was conducted on the period of May 2015 to October 2015.

A total of 260 surveys were returned, achieving 52 % survey response rate. 100 incomplete surveys were exempted from the analysis. Thus, 160 of the returned surveys

were usable responses. The demographic characteristics of the sample population are illustrated in Table 6.1.

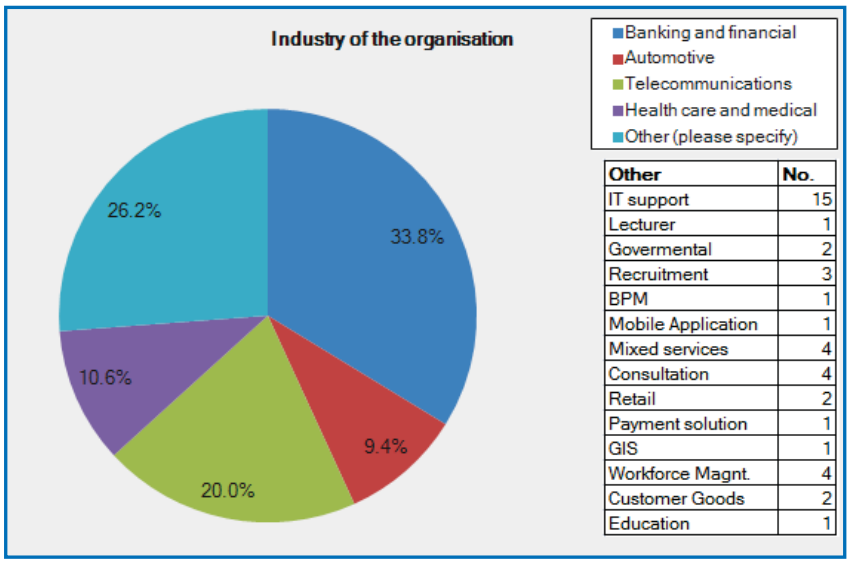
Table 6.1: The characteristics of the sample population

Characteristic	Frequency	Percentage
Industry of the respondent's organisation		
Banking and financial	54	33.8
Telecommunications	32	20
Health care and medical	17	10.6
Automotive	15	9.4
Other	42	26.2
GDAD approach of the organisation		
Best-of-bread and single vendor	69	43.1
Best-of-breed	65	40.6
Single vendor	12	7.5
Other	14	8.8
Open source solutions (OSS)		
Significant portions are OSS	41	25.6
Some portions are OSS	64	40
Few Portions are OSS	44	27.5
No OSS	11	6.9
Job title of respondent (position)		
Developer	22	13.8
Business stakeholder	2	1.2
Team leader/Scrum Master	44	27.5
Analyst	7	4.4
Architect	20	12.5
Project manager	17	10.6
QA/test	3	1.9
Other	45	28.1
GDAD experience of respondent		
Less than 2 years	16	10
2-4 years	62	38.8
5-10 years	64	40
10+ years	18	11.2
Number of teams in the project		
1 team	6	3.8
2 teams	24	15
3-5 teams	65	40.6
6-10 teams	40	25
11-15 teams	16	10

15+ teams	9	5.6
Number of members in the team		
1-3	5	3.1
4-10	41	25.6
11-20	38	23.8
21-30	27	16.9
31-50	20	12.5
51-100	9	5.6
100+	20	12.5

6.2.1 Industry of Respondent's Organisation

As shown in Table 6.1 and Figure 6.1, most of the respondent 33.8% (N=54) worked for banking and financial industry, 20% (N=32) for telecommunication, 10.6% (N=17) for healthcare, 9.4% (N=15) for automotive industry, and 26.2% (N=42) for other industries. Out of the 42 other industries, 15 respondents worked as IT support, 4 for consultation, 4 for mixed services, 4 for workforce management, 3 for recruitment, 2 for government, 2 for customer goods, and only 1 for each of BPM, mobile application, lecturer, payment solution, GIS and education.

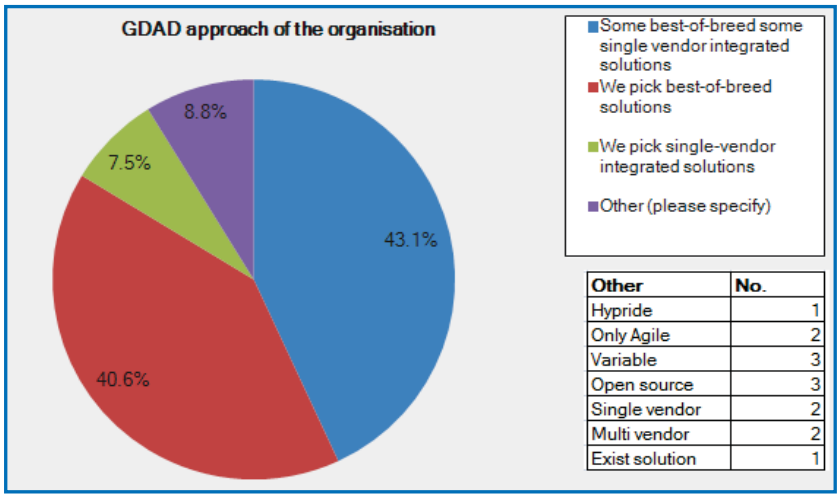


Other	No.
IT support	15
Lecturer	1
Govermental	2
Recruitment	3
BPM	1
Mobile Application	1
Mixed services	4
Consultation	4
Retail	2
Payment solution	1
GIS	1
Workforce Magnt.	4
Customer Goods	2
Education	1

Figure 6.1. Industry of respondent's organisation

6.2.2 GDAD Approach Used by Respondent's Organisation

As shown in Table 6.1 and Figure 6.2, the GDAD approaches used by respondents' organisations to provide solutions were mostly (43.1%, N=69) by some of the best available solution or some single vendor. This was followed by choosing the best-of-breed solution that makes up 40.6% (N=65). 7.5% (N=12) of respondents' organisation used single-vendor integrated solution as the preferred approach of providing solutions to their customers. 8.8% (N=14) had different solution approaches, as shown in Figure 6.2.



Other	No.
Hypride	1
Only Agile	2
Variable	3
Open source	3
Single vendor	2
Multi vendor	2
Exist solution	1

Figure 6.2. GDAD approaches of the respondent's organisation

6.2.3 Open Source Solutions Approach Used by Respondent's Organisation

The sample also showed that most of organisations (40%, N=64) outsourced some of their software products, followed by 27.5% (N=44) of organisations who had only a few portions as outsourcing, 25.6% (N=41) who had significant outsourced portions of their activities, and 6.9% (N=11) did not use outsourcing at all. Figure 6.3 shows the different percentage of respondents' organisations outsourcing activities.

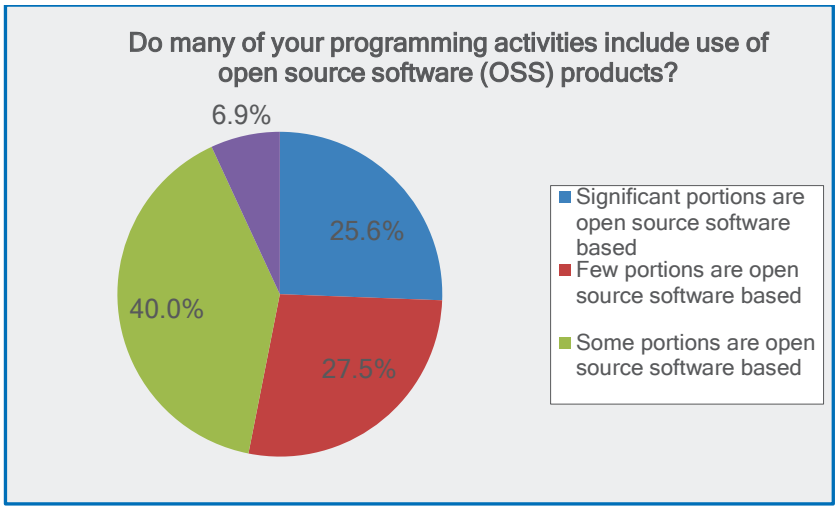
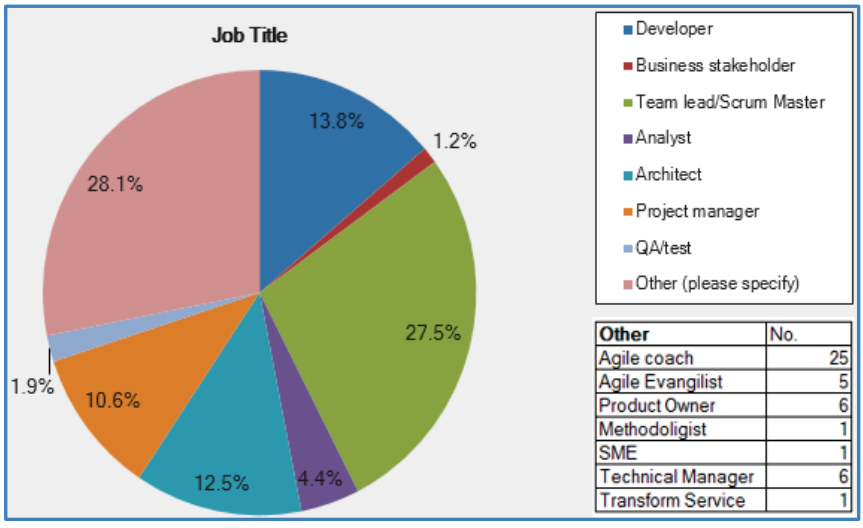


Figure 6.3. Open source activities portion of the respondent's organisation

6.2.4 Respondent's Job Title

As shown in Table 6.1 and Figure 6.4, out of the 160 respondents, 44 were team leader or Scrum Master (27.5%), 22 were developers (13.8%), 20 were architects (12.5%), 17 were project manager (10.6%), 7 were analysts (4.4%), 3 were QA/test (1.9%), 2 were business stakeholders (1.2%), and 45 were of other positions (28.1%). Out of these 45 respondents, 25 were agile coaches (15.6%). 6 respondents were product managers (3.7%), 6 were technical managers (3.7%), and 5 were agile evangelist (3.1%). Only 1 respondent (0.6%) was methodologist, 1 was SME, and 1 was transform service.



Other	No.
Agile coach	25
Agile Evangilist	5
Product Owner	6
Methodoligist	1
SME	1
Technical Manager	6
Transform Service	1

Figure 6.4. Job title for respondent

6.2.5 Experience, number of Teams, and Number of Members in Teams

As shown in Table 6.1, most of the respondents had more than (5-10) years' experience in GDAD (40%), followed by those who had experience of (2-4) years (38.8%), more than 10 years (11.2%), and less than 2 years (10%). The sample showed that most respondents 40.6% (N=65) usually worked with (3-5) teams, 25% (N=40) worked with (6-10) teams, 15% (N=24) worked with 2 teams, 10% (N=16) worked with (11-15) teams, 5.6% (N=9) worked with more than 15 teams, and 3.8% (N=6) were included in 1 team only.

The sample also showed that most teams 25.6% (N=41) included (4-10) members, 23.8% (N=38) included (11-20) members, 16.9% (N=27) included (21-30) members, 12.5% (N=20) included (31-50) members, 12.5% (N=20) included more than 100 members, 5.6% (N=9) included (51-100) members, and 3.1 (N=5) included (1-3) members in the same team.

6.3 Data Screening and Cleaning

Data logging was done to keep track of the sent out, returned, edited, coded, and data entered questionnaires. This was an important step because we adopted a variety of avenues for data collection, and most of respondents had to be communicated with. Without doing this structured, carefully planned, and organised manner of collecting and tracking data, we could have a cumbersomeness and ambiguous task later on. This could consequently lead to misleading and erroroneous data analysis. There was no particular template that was used for tracking purpose. The different codes that were used for each of the variables used for performing the studies are listed in Appendix 6.1.

Data screening and cleaning is an essential step to prepare data for multiple regression analysis (Hair et al. 2010). The data screening and cleaning process also is a crucial step to exclude cases that are outside the defined sample frame (i.e. incomplete response, respondent did not have GDAD experience, respondent that had never used EA, and team size less than 2 individuals), thus it must be executed in precise steps (Hair et al. 2010). The data for this study was collected using two techniques; online (first dataset)

and email (second dataset). Firstly, most of responses were obtained through the web-based survey tool (SurveyMonkey). We used the export capability of the tool to collect data obtained through the tool into Microsoft Excel file (Excel 2007), which enabled a structured view and manipulation of the data and minimised the possibilities of errors occurring during manual data entry. The data were then coded (labelled) according to a naming convention. The coding was done for ease in representation, and analysis using SPSS. Secondly, some responses were collected through emails. These responses were manually added to the *Excel* sheet, which we created for the first data set. The data from these responses were entered into the Excel sheet on a case by case basis with a remark if anything irrelevant was found, as mentioned above. The data were then coded following same coding as for the first dataset. Proofreading of the computerised data file against the original data was done to ensure data accuracy (Tabachnick & Fidell 2007). The full dataset responses (first and second datasets) Excel data sheet was imported as an SPSS data sheet to perform the statistical analysis using SPSS (SPSS v.16). The total number of valid responses entered in the data file was 260 responses (230 web-based instruments and 30 emailed instruments). Out of the 260 responses received, 96 cases which did not match these defined sample frame criteria were deleted. This left 164 responses as usable for further analysis. Table 6.2 tabulates the deleted cases.

Table 6.2: Summary of deleted cases

Reason for Deletion	Number of Cases Deleted	Sample Size Each Deletion
Incomplete case	86	174
Irrelevant respondent GDAD experience	7	167
Irrelevant respondent EA experience	3	164
Irrelevant team size	0	164

6.4 Missing Data Analysis

After screening and verifying the data against the defined sample frame, the next important step was the checking for missing data. Missing data happens when "valid values on one or more variables are not available for analysis" (Hair et al. 2010, p. 49). Missing data is one of the most pervasive problems in data analysis since incomplete questionnaires could bias the results, raise the issue of the generalisability of the result, distort the practical sample size available for analysis, and distort the statistical tests

based on sample size, such as significance level (Hair et al. 2010). This research employed a four-step process proposed by Hair et al. (2010) to deal with missing data due to its completeness as well as its wider use in IS research.

Firstly, the type of missing data (known as missing data patterns) should be determined and determine whether the data can be disregarded or not. Knowing the data type and relationships underlying the missing data is essential and helps maintaining the original distribution of values as closely as possible after any remedy is applied. The literature has identified two types of missing data: "ignorable missing data" and "non-ignorable missing data" (Tabachnick & Fidell 2007). While the ignorable type does not require specific remedies, and can be easily predictable because it is part of the research design (e.g., skipped sections that are not applicable or when the missing data are those observations in a population that are not included when taking a sample), non-ignorable type requires systematic analysis to missing data (Tabachnick & Fidell 2007). One example of ignorable missing data in this research is the use of the 'not decided' option. The current research treats 'not decided' data as 4 on a Likert scale from 1 to 7 which represent a neutral value that does not measure the variables.

Moreover, the 'non-ignorable' missing data cannot be predicted. This type of data can be grouped into two types (Hair et al. 2010), known and unknown missing data processes. The known process allows the researcher to identify the missing data, while unknown processes are less easily identified. For example, missing data due to errors in data entry is classified as a known missing data process while missing data due to the respondents refusing to answer or having no knowledge about certain questions is classified as an unknown process. Most of the 'non-ignorable' missing data occurs due to causes related to the respondent such as when respondents refuse to answer some of the questionnaire items due to their organisation policy or perceptions regarding the sensitive (e.g., their position or their organisations) nature of the questions (Hair et al 2010).

Secondly, assessment of the extents and patterns of missing data that allow the determination of whether the missing data are due to specific cases or variables. The percentage of missing data for variables and cases are calculated to identify if the

amount of missing data per variable or case is low enough to not require any specific treatment or if it affects the results of the study (Hair et al. 2010). The missing data analysis tool from SPSS indicated that out of 4,264 (26 * 164) data points (named as values in SPSS), 97 or 2.3 percent were missing. There were 147 cases (89.6 percent) that had no missing values, and 17 cases that had missing values. These cases contained both non-ignorable missing data (8 cases) and ignorable missing data. Out of 26 variables in total, 9 variables had missing data.

According to Hair et al. (2010) recommendations, the deletion of cases and/or variables can be done according to the following criteria:

- cases with missing data for dependent variables should be deleted since any imputation method applied to these data would cause an artificial increase in the relationship between the dependent and independent variables, and
- cases with percentage of missing data greater than 10 percent should be deleted except when the missing data occurs in a specific non-random fashion (e.g., concentration in a specific set of questions, attrition at the end of the questionnaire).

Although there is no firm guideline on how much missing data can be tolerated for a sample of a given size, a threshold of 5 percent was used, after reviewing the literature (Tabachnick & Fidell 2007). This threshold was chosen because: (1) the SPSS applications are set to display only indicator variables with more than 5 percent missing data, and (2) if only a few data points (5 percent or less) are missing in a random pattern from a large dataset, almost any procedure for handling missing values yields similar results (Tabachnick & Fidell 2007). Table 6.3 shows the four variables with highest percentage of (>5 percent) of missing data.

Table 6.3: Summary statistics for missing data

Variable	Number of Valid Cases	Missing	Mean	Std. Deviation	Missing %
EFFIC4	155	9	3.43	1.524	5.5
EFFIC5	153	11	3.55	1.674	6.7
EFFECT3	151	13	3.48	1.544	7.9
EFFECT4	150	14	3.52	1.536	8.5

As can be seen from Table 6.3, the variables with greater than 5 percent missing data are dependent variables and involved directly in the model, so they were excluded from the sample size. As a result, the remaining data contained 160 cases. However, these cases still contained a certain level of missing data.

Thirdly, the missing data process should be diagnosed whether the missing data are distributed randomly across the cases and the variables. There are two levels of randomness in assessing missing data (Hair et al. 2010): (1) missing data at random (MAR). In this case, data are missing randomly within subgroups but make manifest differences between the subgroups of missing data. (2) Missing completely at random (MCAR). In this case, data are not subject to any underlying process which determines that the data are missing and, therefore, the data do not lead to bias in the observed variable (Allison 2002). MCAR is experienced when the cases with missing data cannot be distinguished from the cases with complete data. When assessing missing data, MAR require particular methods to hold a non-random component while MCAR is sufficiently random so any desired remedy can be used (Hair et al. 2010).

The MCAR test was done by comparing the actual pattern of missing data with what would be expected if the missing data were totally randomly distributed. MCAR occurs when the pattern of missing values does not depend on the data values. The Little's MCAR test (Hill 1997; Little 1988) was used to measure whether data are MCAR. The results of this test are presented in Table 6.4. The Little's MCAR test was found significant at level 0.180 (much larger than 0.05), which indicates a non-significant difference between the observed missing data pattern in the reduced sample and a random pattern (i.e. data are MCAR). With MCAR, no potential bias exists in the pattern of the missing data, and therefore, any remedy for missing data can be used.

Table 6.4: Little's MCAR test results

EM Means						
Mean AEA	Mean EFFIC	Mean EFFECT	Mean TIME	Mean BUDGT	Mean FUNC	Mean QLTY
3.250	3.342	3.480	3.785	3.900	2.950	3.226

Chi-Square = 551.400, DF = 522, Sig. = .180

Finally, the relevant imputation method should be applied for the missing data. The missing data can either be treated through deletion or imputation of missing data (Hair et al. 2010). Since the missing data were found to be completely random, many possible remedies could be used for MCAR data. Such method are pair-wise deletion, case/list-wise deletion, and other imputation methods (e.g., case substitution, expectation maximisation, mean substitution, and cold desk imputation) (Hair et al 2010). In this study, expectation maximisation (EM) was adopted to handle missing data during the measurement development and relationship identification process (Tabachnick & Fidell, 2007).

EM imputation method (Little & Rubin 1987) is suggested in the literature as it produces estimations that are closest to the parameter values (Schafer 1997). EM predicts the missing values of a variable based on its relationship to other variables in the dataset (Hair et al. 2010). 'EM forms a missing data correlation matrix by assuming the shape of a distribution for the partially missing data and basing inferences about missing values on the likelihood under that distribution' (Tabachnick & Fidell 2001, p. 68). Moreover, to produce high quality imputations for a particular variable both variables that are potentially related to the missing values need to be enclosed (i.e. the imputed variable and variables that are potentially related to the imputed variable) (Schafer 1997). Therefore, EM method represents the original distribution of values and gives consistent and unbiased estimates of correlations and covariances (Hair et al. 2010). EM imputation method is supported in SPSS v.16 and was used in this research. SPSS produced a new data sheet with imputed missing values which were then used for further analysis.

Although the dataset had been carefully treated for missing data, the data values themselves still required further analysis on their own characteristics. The next section presents the outlier treatment of the data.

6.5 Outliers Testing

After screening and checking the data for missing values, the next important step was checking for outliers' values. Outliers are observations with scores that have a unique combination of characteristics and are substantially different from the rest of observations (Hair et al. 2010). Examining data to detect outliers is important since they can potentially bias the mean, inflate the standard deviation, change the results of the analysis, and lead to poor model fit (Tabachnick & Fidell 2007). Outliers can be either beneficial or problematic to the analysis. From the positive angle, beneficial outliers may represent indications of characteristics of the population that would not be discovered by normal analysis. From the negative point of view, problematic outliers do not represent the population, are linked to the issues related to variables' normality and skewness, and can seriously distort the analysis. Therefore, the analysis for outliers not only checks for the presence of outliers in the data but also the direction of the impact on results, whether harmful or helpful (Hair et al. 2010).

Cases that have scores greater than three standard deviations beyond the mean may be considered as outliers (Kline 2015). In order to detect such extreme deviations in this research, the entire scores of all 26 variables from all cases were converted into standardised z-scores. Any cases with an absolute z-score (|z|) value > 3.29 or < -3.29 $p < 0.01$ were considered potential outliers (Tabachnick & Fidell 2007). The number of outliers should not be greater than approximately one percent for any variable (Tabachnick & Fidell, 2007). In this study, two variables scored 3.29 < z-score < -3.29 (see Table 6.5), accounting for 1.25%; this was not excessive compared with the acceptable level of one percent. Moreover, to ensure the outliers did not significantly result in distortion to the data, the differences between the mean and the 5% trimmed mean of each variable was examined (i.e. Δ Mean). The '5% trimmed mean' is the mean calculated from a set of cases where the cases score five percent in the top and the bottom are removed (Pallant 2001). By convention, a large Δ Mean (i.e. > 0.20) indicates that the outliers may cause a problem to the data set (Pallant 2001). In this study, all Δ Means were relatively small compared with 0.20, ranging from 0.020 to 0.090 (see Table 6.5). These results indicate that the detected outliers do not have any problem to the data set. Accordingly, all 160 cases were retained for further statistical analyses.

Table 6.5: Normal distribution and outlier tests for reflective indicators

Variable	Mean	5% Trimmed Mean	Std. Dev.	Z-skew	Z-kurt	Δ Mean (Mean-5% Trimmed Mean)	Z-Score (Min to Max)
EFFIC1	3.130	3.060	1.514	0.733	-0.175	0.070	-1.408 – 2.556
EFFIC2	3.320	3.280	1.506	0.516	-0.628	0.040	-1.539 – 2.443
EFFIC3	3.310	3.260	1.497	0.511	-0.336	0.050	-1.544 – 2.463
EFFIC4	3.440	3.400	1.524	0.449	-0.530	0.040	-1.599 – 2.337
EFFIC5	3.520	3.470	1.648	0.433	-0.722	0.050	-1.532 – 2.108
EFFECT1	3.330	3.270	1.565	0.652	-0.509	0.060	-1.489 – 2.344
EFFECT2	3.560	3.520	1.561	0.485	-0.647	0.040	-1.641 – 2.202
EFFECT3	3.470	3.410	1.529	0.669	-0.195	0.060	-1.614 – 2.309
EFFECT4	3.54	3.490	1.533	0.579	-0.230	0.050	-1.659 – 2.254
TIME1	3.710	3.670	1.681	0.417	-0.892	0.040	-1.610 – 1.959
TIME2	3.860	3.840	1.710	0.308	-0.952	0.020	-1.673 – 1.834
BUDGT1	3.850	3.800	1.567	0.570	-0.618	0.050	-1.818 – 2.010
BUDGT2	3.920	3.880	1.633	0.439	-0.796	0.040	-1.787 – 1.887
FUNC1	2.880	2.790	1.196	1.093	1.905	0.090	-1.567 – **3.448**
FUNC2	2.890	2.830	1.192	0.739	0.898	0.060	-1.583 – **3.449**
FUNC3	3.100	3.020	1.323	0.968	1.089	0.080	-1.587 – 2.947
QLTY1	3.010	2.940	1.261	0.807	0.396	0.070	-1.590 – 3.166
QLTY2	3.460	3.400	1.414	0.625	0.043	0.060	-1.741 – 2.502
QLTY3	3.230	3.170	1.370	0.718	0.418	0.060	-1.628 – 2.750

6.6 Testing the Underlying Assumptions of Multiple Regression

As mentioned before, this study will use the nonparametric analysis method (i.e. partial least squares structural equation model [PLS-SEM]) to test the model since the construct "Agile EA" was modelled as formative. PLS-SEM, different from covariance-based structural equation model (CB-SEM), does not require the checking for the underlying assumptions such as normality and linearity. Nevertheless, checking for normality for example is important "to verify that the data are not too far from the normal as extremely non-normal data prove problematic in the assessment of the parameters' significances. Specifically, extremely non-normal data inflate standard errors obtained from bootstrapping and thus decrease the likelihood some relationships will be assessed significant" (Hair et al. 2014, p.54). Therefore, this study will examine the four most important tests for the assumptions of multivariate analysis are:

multivariate normality, linearity, homoscedasticity, and multicollinearity. These tests are applied for reflective constructs only (i.e. EFFIC, EFFECT, TIME, BUDGT, FUNC, and QLTY). Satisfying these assumptions, data will result in a precise reflection of reality.

6.6.1 Normality

Normality is the most important fundamental assumption in multivariate analysis (Hair et al. 2010; Tabachnick & Fidell 2007). Normality refers to the shape (a bell-shaped curve and the standard normal distribution has a mean of 0 and a standard deviation of 1) of the data distribution for a variable and its correspondence to the normal distribution (Hair et al. 2010). Multivariate analysis requires multivariate normality assumption for both the individual variables and the combinations of these variables. The violation of normality assumption can have two dimensions: the shape of the distribution and the sample size. Non-normal data may result in underestimated standard errors and inflated goodness of fit statistics (Tabachnick & Fidell 2007). There are two types of normality: univariate and multivariate (Hair et al. 2010). While 'univariate normality' refers to the degree to which the data of a specific variable is normally distributed, 'multivariate normality' refers to the normal distribution of the data of a combination of more than one variable (Hair et al. 2010).

Multivariate normality can be addressed when conducting the model assessment using the Structural Equation Modelling (SEM) technique. In general, the assessment of normality can be carried out visually (observe and judge how well a variable's data histogram corresponds to a bell-shaped curve) or statistically (using "kurtosis" and "skewness" values) (Hair et al. 2010). Researchers commonly adhere to statistical assessment using "skewness" and "kurtosis", which are considered to be two important components of normality (Tabachnick & Fidell, 2007).

Skewness describes the symmetrical balance of the distribution (i.e. a skewed variable is a variable whose mean is not in the centre of the distribution) (Hair et al. 2010). Skewness can either be positive (the distribution is unbalanced and shifted to the left) or negative (the distribution is unbalanced and shifted to the right). Kurtosis however,

provides information about the "peakedness" or "flatness" of a distribution compared with the normal distribution (Hair et al. 2010). A negative kurtosis indicates that the distribution is flatter than the normal distribution while a positive kurtosis indicates a higher peak than the normal distribution (Hair et al. 2010). Theoretically, the perfect distribution value of both skewness and kurtosis is zero (uncommon occurrence in social research). Both skewness and kurtosis values should fall between -2.00 and +2.00 to have a normal distribution (Hair et al. 2010). Not normally distributed in terms of kurtosis and skewness indicates a violation of the assumption of normality. However, the impact of violation depends on the sample size, also. If the sample is small, significant departure from normality can have a greater impact, whilst in a large sample size (e.g., 200 or more), the same impact may be negligible (Hair et al. 2010).

Analysis of normality using the visual method can be done by comparing the normal distribution that forms a straight diagonal line with the line presenting the actual data distribution. If the line follows the diagonal, the distribution of data is normal. Any departure from a normal distribution will result in a deviation of the line from the diagonal. Analysis of normality using the statistical values of the "skewness" and the "kurtosis" are measured by standardised z-scores, which are named "z-skewness" and "z-kurtosis" respectively. As a rule of thumb, values for skewness and kurtosis divided by the standard error should be within -1.96 and +1.96 for 0.05 errors level, or more linearly, within -2.58 and +2.58 for 0.01 error level for normally distributed variables (Hair et al. 2010). The more lenient -3 to +3 range is also widely used in the literature (Tabachnick & Fidell 2007).

The current research employed the statistical test of normality. As shown in Table 6.5, the results of normal distribution tests indicated that the absolute values of skewness and kurtosis of all variables ranged from 0.238 to 1.093 and from -0.952 to 1.905 respectively, which fall within the aforementioned recommended ranges of -1.96 to +1.96. These results indicate that all variables were normally distributed.

6.6.2 Linearity

Linearity refers to a consistent slope of change that represents the relationship between the dependent and independent variables (Hair et al. 2010). If a relationship is nonlinear, the statistics will not only underestimate the strength of the relationship but also cause inaccurate predictions, especially when the result is generalised beyond the range of the sample. As for the previous tests (i.e. normality), linearity test can be done using graphical or statistical method (Tabachnick & Fidell 2001). In this study, we used the graphical method to test the linearity. This was done by analysing the residuals and partial regression scatter plots. We analyse the scatter plot of the actual standardised values (ZRESID) of the dependent variables (e.g., EFFECT) against the predicated residual values (ZPRED) of the independent variable (e.g., EFFIC) as the most common test to check for linearity and homoscedasticity (Hair et al. 2010). If there is a linear relationship between two variables, the scatter plot will be oval (Hair et al. 2010). As shown in Figure 6.5, the residuals scatter plot shows that the points are randomly distributed, which means that the scatter plot does not exhibit non-linear patterns (Hair et al. 2010). This indicates that the assumption of linearity for variables has been met.

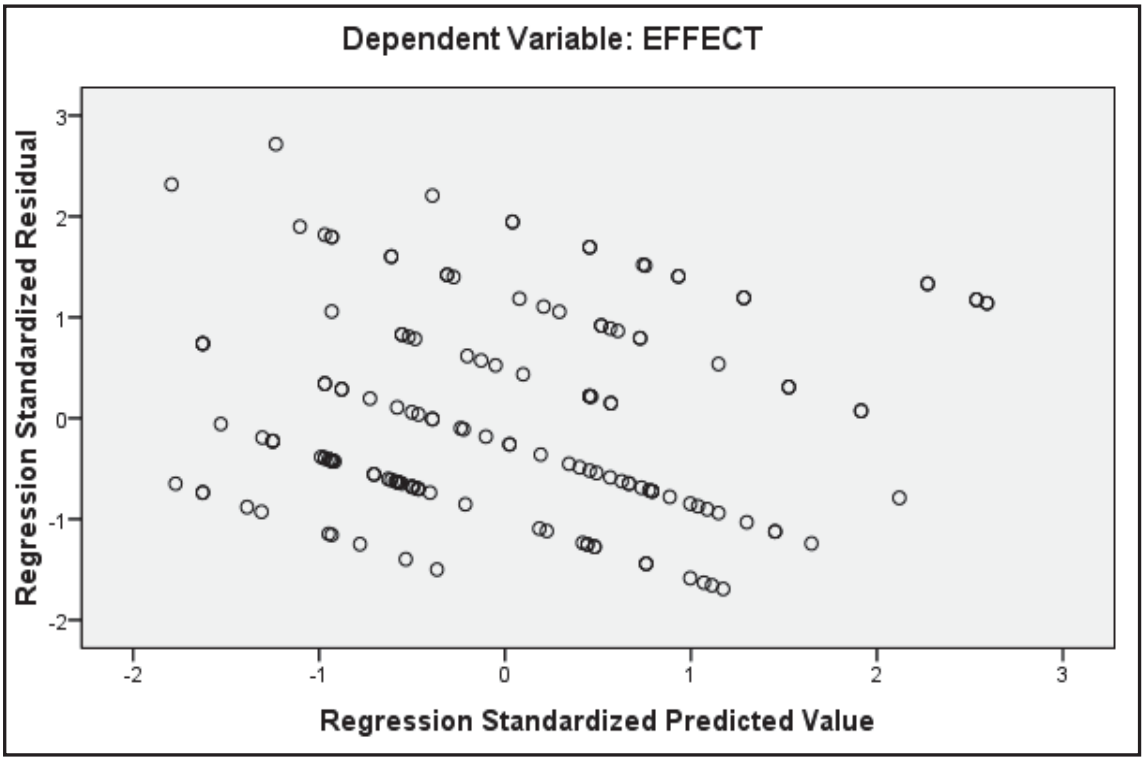


Figure 6.5. Scatter plot: EFFIC vs. EFFECT

6.6.3 Homoscedasticity

The test for homoscedasticity, which is also known as the test for homogeneity of variance, involves checking if the variance of the dependent variable being explained in a dependence relationship exhibits a similar size across the range of predictor variable(s) rather than being concentrated in only a limited range of the independent variables (Hair et al. 2010). To conduct the homoscedasticity test, the variance of the dependent variable values is compared at each value of the independent variable. If the variance values are relatively equal, it indicates that the relationship between the dependent variable and the independent variable exists. The two most common sources of heteroscedasticity are the type of the variables and the skewed distribution of one or both variables. Homoscedasticity is strongly related to the assumption of a normal distribution such that when the assumption of multivariate normality is met, the relationships between variables are homoscedastic (Tabachnick & Fidell 2007). Homoscedasticity is evaluated for pairs of variables and can be conducted through either statistical or graphical methods. This research applied a graphical method. As for linearity, homoscedasticity was tested by analysing the residual plots of the actual standardised values (ZRESID) of the dependent variable against the predicated residual values (ZPRED) of the independent variables. As also shown in Figure 6.5, the data were randomly dispersed throughout the scatter plot, which means that the data does not violate the homoscedasticity test.

6.6.4 Multicollinearity

Multicollinearity occurs when two or more independent variables are highly correlated, which will result in logical (i.e. when redundant variables are included while they are not needed in the same analysis) and statistical issues (Hair et al. 2010). This case occurs when the independent variables measure the same thing, which leads to redundant measures. Items that measure the same construct must be correlated, however; correlations should not be greater than 0.9 between any two items or otherwise statistical problems will happen (Tabachnick & Fidell 2007). In this research, the statistical analysis for the multicollinearity problem was conducted by assessing the item-item correlation matrix using Pearson values (refer to Appendix 6.2). No multicollinearity issue was identified.

6.7 Exploratory Factor Analysis

Having established the profile and the underlying assumptions of the sample population, the next step was to use exploratory factors analysis (EFA) to determine if the large number of variables included in the data could be reduced to a smaller number of representative factors. Also, EFA results were used to test the validity and the unidimentionality of the measures used in this study.

Since the latent constructs are measured by one or more variables, it is important to test the factorial validity that identifies if the underlying structure among the variables extracted from the data is consistent with those proposed by the research model. So, factorial validity ensures that the construct is independent of its theoretical connections to ensure that the variables are sufficiently intercorrelated to produce representative factors (Straub et al. 2004).

Factorial validity is measured usually by multi-trait multi-method (MTMM) matrix or factor analysis (Lewis et al. 2005). While MTMM is preferred within the field when more than one research method is used, factor analysis appears to be the more commonly used technique when a single method is employed (Straub et al. 2004). Since a single method (i.e. survey) is used, factorial validity was established through EFA that helps to derive the initial set of factors for the reflective construct (Lewis et al. 2005). The EFA was conducted with the following extraction condition: Minimum eigenvalue of 1, principal component analysis, and using the VARIMAX rotation method.

First, an important concern in EFA is sample adequacy (Hair et al. 2010). As it was discussed in Section 5.6.5, the sample size of the study of 160 is above the minimum size of 84, according to Cohen power analysis (Cohen 1988). This guaranteed a reliable EFA procedure (Straub et al. 2004).

Second, results from the first step provide a basis to proceed in examination of factor analysis adequacy. The applicability of using EFA was verified by checking the values of Kaiser-Meyer-Olkin (KMO) test (sampling adequacy test), Barlett's sphericity test, and analysis of correlation to assess the strength of the intercorrelation between items

(Hair et al. 2010). First, the KMO test indicated an acceptable value of 0.5 or greater for all reflective latent constructs. Second, the Bartlett's sphericity test that provides evidence that the correlation matrix has significant correlations ($p<0.001$) among at least some of the variables, and the original correlation matrix is an identity matrix indicated that the correlation matrix generated from the data was different from the identity matrix. Third, the correlations should be greater than 0.30 (Hair et al. 2010). Table 6.6 shows that the values of KMO were above 0.5 and Bartlett's sphericity tests were significant for all the six constructs. It also shows that all reflective items communalities were above the threshold value 0.5 (Hair et al. 2010) and items correlations were above 0.3 (see Appendix 6.2).

Third, the eigenvalue (>1) and scree plot were investigated to identify the number of components in the scale. Table 6.7 shows that one component has been identified for each construct, with eigenvalue greater than 1, and the scree plots for each construct identify only one component for each construct. As a final point, factor loading was examined for the scale items. In general, loading value less than 0.4 is assumed low. Items that have loadings below the threshold should be dropped from further analysis (Hair et al. 2010). The test of unidimensionality provides evidence that each summated scale should consist of items correlated and loaded highly in a single factor. Table 6.7 shows that the loading values were above the threshold level. This indicates that all items for each construct are unidimensional.

Table 6.6: KMO and Bartlett's sphericity tests, and communality matrix for reflective constructs

<table>
<tr><th>KMO and Bartlett's sphericity</th><th>Communality Matrix</th></tr>
<tr><td colspan="2">Communication efficiency (EFFIC)</td></tr>
<tr><td>

KMO and Bartlett's Test

Kaiser-Meyer-Olkin Measure of Sampling Adequacy.		.840
Bartlett's Test of Sphericity	Approx. Chi-Square	687.058
	df	10
	Sig.	.000

</td><td>

All items correlations were above 0.3

Communalities

	Initial	Extraction
EFFIC1	1.000	.777
EFFIC2	1.000	.776
EFFIC3	1.000	.871
EFFIC4	1.000	.831
EFFIC5	1.000	.659

Extraction Method: Principal Component Analysis.

</td></tr>
<tr><td colspan="2">Communication effectiveness (EFFECT)</td></tr>
<tr><td>

KMO and Bartlett's Test

Kaiser-Meyer-Olkin Measure of Sampling Adequacy.		.807
Bartlett's Test of Sphericity	Approx. Chi-Square	526.097
	df	6
	Sig.	.000

</td><td>

All items correlations were above 0.3

Communalities

	Initial	Extraction
EFFECT1	1.000	.798
EFFECT2	1.000	.784
EFFECT3	1.000	.832
EFFECT4	1.000	.857

Extraction Method: Principal Component Analysis.

</td></tr>
<tr><td colspan="2">On-Time completion (TIME)</td></tr>
<tr><td>

KMO and Bartlett's Test

Kaiser-Meyer-Olkin Measure of Sampling Adequacy.		.500
Bartlett's Test of Sphericity	Approx. Chi-Square	276.071
	df	1
	Sig.	.000

</td><td>

All items correlations were above 0.3

Communalities

	Initial	Extraction
TIME1	1.000	.955
TIME2	1.000	.955

Extraction Method: Principal Component Analysis.

</td></tr>
<tr><td colspan="2">On-Budget completion (BUDGT)</td></tr>
<tr><td></td><td>All items correlations were above 0.3</td></tr>
</table>

KMO and Bartlett's Test

Kaiser-Meyer-Olkin Measure of Sampling Adequacy.		.500
Bartlett's Test of Sphericity	Approx. Chi-Square	268.736
	df	1
	Sig.	.000

Communalities

	Initial	Extraction
BUDGT1	1.000	.952
BUDGT2	1.000	.952

Extraction Method: Principal Component Analysis.

Functionality (FUNC)

All items correlations were above 0.3

KMO and Bartlett's Test

Kaiser-Meyer-Olkin Measure of Sampling Adequacy.		.669
Bartlett's Test of Sphericity	Approx. Chi-Square	179.203
	df	3
	Sig.	.000

Communalities

	Initial	Extraction
FUNC1	1.000	.817
FUNC2	1.000	.619
FUNC3	1.000	.751

Extraction Method: Principal Component Analysis.

Quality (QLTY)

All items correlations were above 0.3

KMO and Bartlett's Test

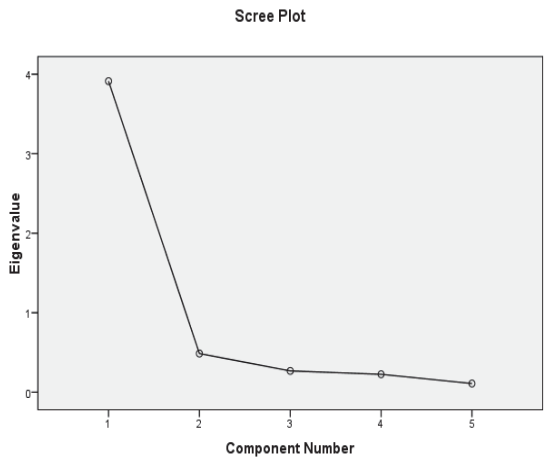
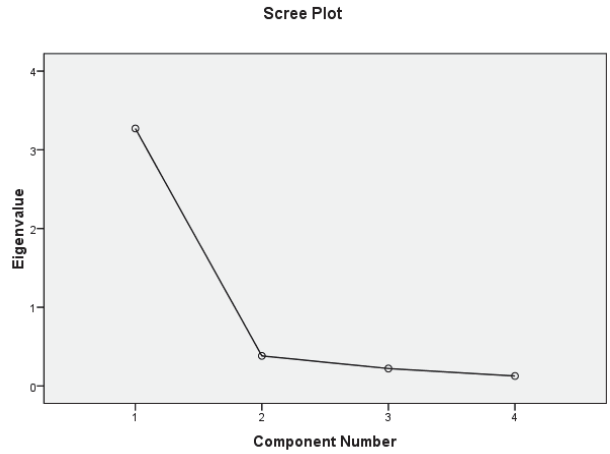
Kaiser-Meyer-Olkin Measure of Sampling Adequacy.		.675
Bartlett's Test of Sphericity	Approx. Chi-Square	195.677
	df	3
	Sig.	.000

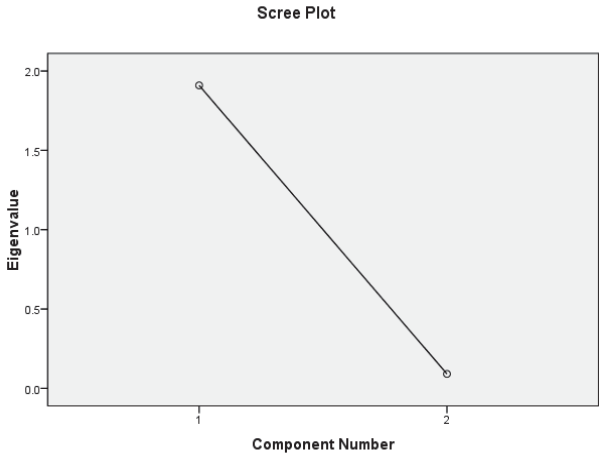
Communalities

	Initial	Extraction
QLTY1	1.000	.663
QLTY2	1.000	.744
QLTY3	1.000	.834

Extraction Method: Principal Component Analysis.

Table 6.7: Scree plot and component matrix for reflective constructs

<table>
<tr><th>Scree Plot</th><th>Component Matrix</th></tr>
<tr><td colspan="2">Communication efficiency (EFFIC)</td></tr>
<tr><td> 
</td><td>Eigen value = 3.913, % variance = 78.262
Component Matrix[a]
<table>
<tr><th></th><th>Component</th></tr>
<tr><th></th><th>1</th></tr>
<tr><td>EFFIC1</td><td>.881</td></tr>
<tr><td>EFFIC2</td><td>.881</td></tr>
<tr><td>EFFIC3</td><td>.933</td></tr>
<tr><td>EFFIC4</td><td>.912</td></tr>
<tr><td>EFFIC5</td><td>.812</td></tr>
</table>
Extraction Method: Principal Component Analysis.
a. 1 components extracted.</td></tr>
<tr><td colspan="2">Communication effectiveness (EFFECT)</td></tr>
<tr><td> 
</td><td>Eigen value = 3.270, % variance = 81.762
Component Matrix[a]
<table>
<tr><th></th><th>Component</th></tr>
<tr><th></th><th>1</th></tr>
<tr><td>EFFECT1</td><td>.893</td></tr>
<tr><td>EFFECT2</td><td>.885</td></tr>
<tr><td>EFFECT3</td><td>.912</td></tr>
<tr><td>EFFECT4</td><td>.925</td></tr>
</table>
Extraction Method: Principal Component Analysis.
a. 1 components extracted.</td></tr>
<tr><td colspan="2">On-Time completion (TIME)</td></tr>
</table>



Eigen value = 1.909, % variance = 95.462

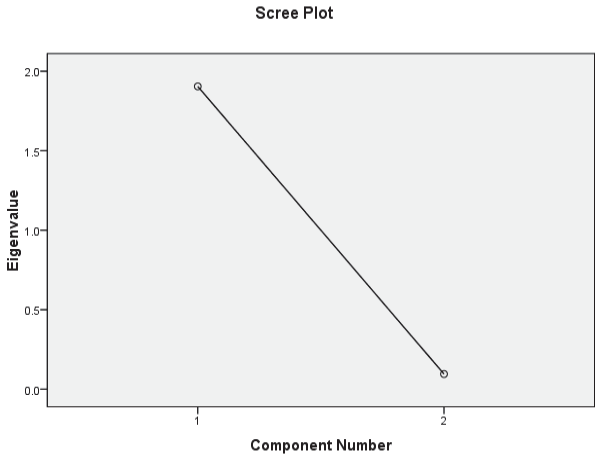
Component Matrix[a]

	Component
	1
TIME1	.977
TIME2	.977

Extraction Method: Principal Component Analysis.

a. 1 components extracted.

On-Budget completion (BUDGT)



Eigen value = 1.905, % variance = 95.234

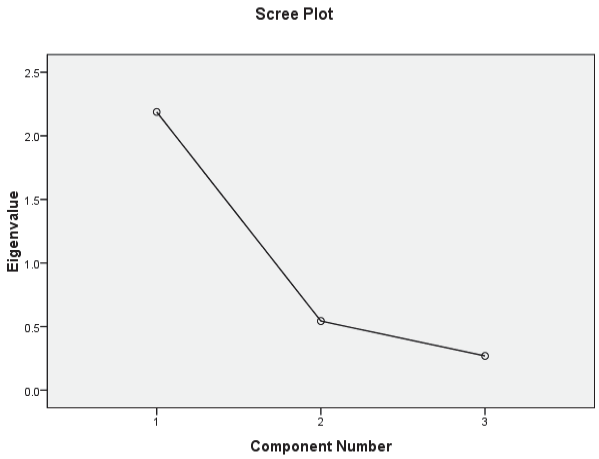
Component Matrix[a]

	Component
	1
BUDGT1	.976
BUDGT2	.976

Extraction Method: Principal Component Analysis.

a. 1 components extracted.

Functionality (FUNC)



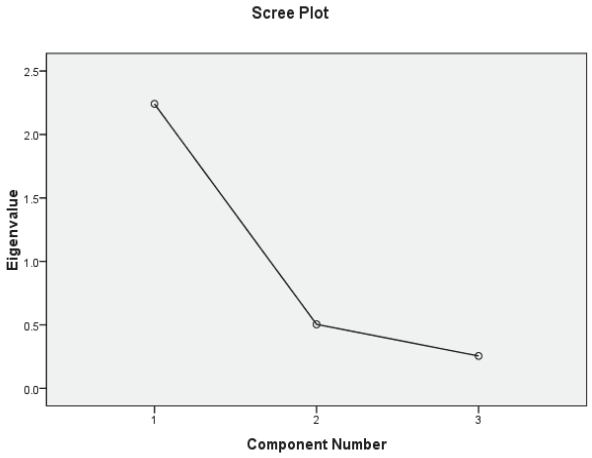
Eigen value = 2.188, % variance = 72.921

Component Matrix[a]

	Component
	1
FUNC1	.904
FUNC2	.787
FUNC3	.867

Extraction Method: Principal Component Analysis.

a. 1 components extracted.

Quality (QLTY)	
	Eigen value = 2.241, % variance = 74.703 **Component Matrix[a]** Component: 1 QLTY1 .814 QLTY2 .862 QLTY3 .913 Extraction Method: Principal Component Analysis. a. 1 components extracted.

6.8 Generalisability Testing

Data was collected from different locations in the world (no specific context or country was used). To ensure that it is possible to generalise the research findings from the study sample to the population, two data generalisability tests were conducted. We tested for bias due to collecting method and also for non-response bias. These tests are discussed in the following sections.

6.8.1 Common Method Bias

Common method biases may occur when data is collected via only one method (survey in the case of this study), or via the same method but only at one point in time (Straub et al. 2004). This may result in a variance that the items have in common with each other due to the data collection method rather than the hypothesised relationships between constructs or between items and constructs (Malhotra et al. 2006). The method variance causes a systematic measurement error and by either inflating or deflating the observed relationships between the theoretical constructs (Podsakoff et al. 2003). As the research involved a self-report survey design, it has the risk of common method bias. However, it has been questioned that the method alone causes systematic variance (Spector 2006). To assess whether or not potential common method bias was a significant issue, we

performed Harman's one-factor (Podsakoff & Organ 1986) statistical test to the six reflective constructs (i.e. EFFIC, EFFECT, TIME, BUDGT, FUNC, and QLTY), as shown in the last raw data of Table 6.8, as the most widely used test reported in the literature (Podsakoff et al. 2003). Harman's test examines the unrotated factor solution of all reflective variables of the study (i.e. 19 items) in an exploratory factor analysis. If the test shows a substantial amount of common method variance (i.e. greater than 50 percent), two techniques can be applied (Podsakoff et al. 2003): (1) a single factor will emerge from the factor analysis, or (2) one general factor will account for the majority (i.e. ≥ 50 percent) of covariance among the variables.

The EFA was conducted according to two conditions: number of factors was extracted to 1 and no rotation method was used. The analysis revealed that there was no single factor which explained a substantial amount of variance (the most covariance explained by one factor is only 31 percent), which provides evidence that common method biases do not pose a significant threat to the measurement validity of this study. Moreover, the items that represent one construct were distributed and not grouped together (Gregor & Klein 2014) in the instrument to minimise the common method bias.

Table 6.8: Harman's one-factor test for reflective constructs

Component	1	2	3	4	5	6
Eigenvalue	4.052	3.572	3.365	2.297	1.743	1.107
Variance Extracted	21.325	18.800	17.710	12.091	9.176	5.829
Cumulative Variance (%)	21.325	40.125	57.834	69.925	79.101	84.930
Unrotated Variance (%)	31.350	19.854	13.707	9.927	5.940	4.152

6.8.2 Non-Response Bias

The discrepancies between researcher and respondent perceptions of the meanings of items' statements can cause response bias for individual items across the sample (Lewis et al. 2005). Non-response may result in sample bias since the respondents' answers may differ from the potential answers of those who did not respond (Lewis et al. 2005). Therefore, non-response bias will cause low accuracy of estimations when the research findings are generalised to the population. The current research had a response rate of 52 percent, which resulted in a final sample size of 160 cases. It was important to

conduct the non-response bias test in the current research (Sivo et al. 2006). One widely used method for testing non-response bias is to compare the data that is collected early to the late collected ones (Sivo et al. 2006). The late respondents group is assumed to have similar characteristics to the early group. Therefore, comparing the characteristics of late respondents to early respondents will reveal if non-response bias exists (Sivo et al. 2006).

The sample was segregated into two subsamples, according to the collected data (i.e. one representing the group collected between May 2015 to July 2015, and the second group of the period between August 2015 and October 2015). The first group had 95 cases (59.4 percent) while the second group had 65 cases (40.6 percent). Each returned response was recorded with the date it was received.

The two samples independent t-test at ($p<0.05$) percent significance level was used to compare the two subsamples. The result of the test on the mean score of the construct is displayed in Table 6.9. The full result on all variables is provided in Appendix 6.3. The results showed no significant difference between the two sample groups and thus non-response bias should not cause major problems for generalisability. Therefore, the data for both groups could be combined.

Table 6.9: Non-response bias test

	F	Sig.	Late Response	Early Response	Mean Difference	Std. Error Difference	t	Sig. (2-tailed)
Mean AEA	0.001	0.980	21.630	23.515	1.885	1.334	1.412	0.160
Mean EFFIC	0.326	0.569	16.641	16.905	0.443	1.095	0.405	0.686
Mean EFFECT	0.497	0.482	13.815	13.968	0.153	0.903	0.169	0.866
Mean TIME	0.010	0.919	7.569	7.568	-0.000	0.534	-0.002	0.999
Mean BUDGT	0.074	0.785	7.523	7.936	0.413	0.503	0.822	0.412
Mean FUNC	0.213	0.645	8.907	8.831	-0.076	0.511	-0.149	0.882
Mean QLTY	1.585	0.210	9.692	9.705	0.012	0.564	0.023	0.982

6.9 Chapter Summary

The objective of this chapter was to prepare the data collected in the survey for final analysis. Therefore, the data were put through a systematic process of screening and cleaning, missing values, outliers, normality, linearity, homoscedasticity, multicollinearity, non-response bias, and common method bias. It is important to mention here that the underlying assumptions (normality, linearity, homoscedasticity, and multicollinearity) were applied only to reflective constructs. Although this study used PLS analysis that does not require these assumptions to be met, ensuring that the data were not very distorted was supportive to the findings of this study. The summary of all tests results is shown in the following table (Table 6.10). A total of 100 cases were deleted after the data screening and cleaning procedures, which leaves a final sample size consisting of the data from 160 respondents. Since there are 7 formative indicators of agile EA, PLS analysis was, mainly, used as the analytic method instead of other CB-SEM methods that cannot deal with formative constructs.

Table 6.10: Summary of data preparation

Data Preparation Step	Heuristic Criteria	Result	Action	Number of Cases
Data collection				260
Data screening	Meet sample frame requirement	86 incompletes and 10 cases outside sample frame	Removed the 96 cases	164
Missing data	No missing data on dependent variable greater than 5 percent	4 cases missing data on dependent variables > 5%. Mean substitution was adopted for missing values. MCAR test was found 0.180 (much larger than 0.05 significance level)	Removed the 4 cases	160
Outliers testing	Statistical z-score values should be within -3.29 and 3.29	2 variables had z-score >3.29 only, which does not discriminate the results	No cases were removed	160
Normality	Statistical values for the skewedness and kurtosis within -1.96 and 1.96	All reflected variables were normally distributed and scored in the range of (-1.96<z-score<1.96) for both kurtosis and	Used nonparametric analysis methods (e.g., PLS)	160

		skewness		
Linearity	Graphical method: scatter plot	Values were randomly scattered. No violation was identified to linearity	PLS used	160
Homoscedasticity	Graphical method: scatter plot	Values were randomly scattered. No violation was identified to linearity	PLS used	160
Multicollinearity	Item-item correlation matrix (correlation higher than 0.9)	No extreme value of correlation	PLS used	160
EFA	Eigen value>1, KMO>0.5, Barlett's test ($p<0.001$), variance>60%, Communality>0.5	All these conditions were met for all the six reflective constructs	All reflective constructs were ready for PLS analysis	160
Common method bias	Harman's single factor test < 50%	one general factor accounts for 31.350 percent of the total variance	All reflective constructs were ready for PLS analysis	160
Non-response bias	Independent sample t-test (mean difference not statistically significant)	Independent sample t-test was found not significant and much larger than 0.05 to all variable means	All constructs were ready for PLS analysis	160

Chapter 7 discusses the quantitative findings of this study. It discusses the validation of both reflective and formative measurement models. It also discusses the PLS-SEM findings; hypotheses testing, predictive power and effect size of the model, and the mediating effect. Furthermore, it discusses the second-order model and the goodness-of-fit of the research model.

Chapter 7 Survey Findings

In Ch.6, preparation process was conducted on the collected data. The preparation process included data cleaning, missing data analysis, and testing for outliers. After applying the screening process, some data (i.e. data outside the sample frame) were excluded, and then the remaining data accepted for further analysis. The reflective constructs were investigated against the multivariate underlying assumption (normality, linearity, homoscedasticity, and multicollinearity). Also, EFA was conducted on the reflective constructs to verify the number of factors for each construct and to assess the validity and the unidimentionality of the measures used in the study. Finally, the data was tested against common method and non-response biases to ensure that our results can be generalised for the study population. Having the data prepared, now we can proceed with PLS-SEM analysis.

7.1 Introduction[5]

This chapter continues with the data analysis in two distinct stages: assessment of measurement model and assessment of SEM (Hair et al. 2010). In the first stage, the assessment and validation of the measurement model involved assessment of the reliability, convergent validity, and discriminant validity, as pre-conditions for testing the overall research model. The validity and reliability tests are conducted to ensure the rigour of the measurement model since the overall research model cannot be tested unless the measurement properties of its constructs are reliable and valid. Validity and reliability tests were conducted differently for both reflective and formative constructs. For Reflective constructs, validity was tested by convergent validity and disrcriminant validity. Moreover, the reliability was tested by internal consistency of the measures, composite reliability, and indicator reliabilities. For Formative constructs, reliability

5 The findings of this chapter appeared in Alzoubi & Gill 2017b

was assessed by testing the collinearity among the indicators and the validity was assessed by testing the indicator validity. In the second stage, the assessment of the structural model validity involved testing the model's predictive power or variance explained (R^2), the effect size (f^2), the predictive relevance validity by testing the cross-validated redundancy (Q^2), and the path coefficient (β) significance between constructs. Moreover, extra analysis was conducted to study the mediation effect between exogenous (agile EA) and endogenous (on-time completion, on-budget completion, functionality, and quality) latent variables. In addition, a modified second-order PLS model was tested to examine the effects of agile EA, communication efficiency, and communication effectiveness on overall GDAD software performance. As was discussed in Ch.5, this study used PLS-SEM approach, particularly used SmartPLS 3.0 software package (Ringle et al. 2015) to test the structural equation model and the proposed hypotheses.

This chapter is organised as follows. The rationale of using PLS-SEM analysis is presented in Section 7.2. Section 7.3 discusses the assessment of the measurement model. Section 7.4 discusses the structural model assessment. Section 7.5 discusses the results of the second-order model. Section 7.6 discusses the model "goodness-of-fit" notion. Finally, the chapter summary is presented in Section 7.7.

7.2 PLS-SEM Approach

SEM is a statistical technique that simultaneously tests and estimates causal relationships between dependent and independent constructs (Gefen et al. 2000). Since the goal of this study is to test the relationships between agile EA, GDAD active communication, and GDAD performance, SEM approach is an appropriate approach to test these relationships. One of the major strengths of SEM approaches is the support to latent variables, which are unobservable (cannot be measured directly) constructs that are hypothetically created by researchers in order to understand a research area such as intentions, impacts, and satisfaction (Hair et al. 2014). SEM consists of two sub-models: the measurement model (outer model) and the structural model (inner model). The measurement models are devoted to defining each of the latent variables within the SEM. The measurement model represents the relationship between the latent variables

and their observable indicators. On the other hand, SEM represents the causal relationships between the latent variables, where the independent latent variables are called exogenous variables and the dependent latent variables are called endogenous variables (Hair et al. 2014).

In general, there are two types of indicators: reflective indicators and formative indicators (Chin 1998). Reflective indicators are caused by their latent variables, measure the same underlying phenomenon, and change according to changes to the latent variables (Chin 1998). Therefore, all reflective indicators must correlate positively (Chin 1998). By contrast, formative indicators are not expected to have covariation within the same latent variable and they are causes of, rather than caused by, their latent variables (Chin 1998; Petter et al. 2007). Therefore, formative indicators do not necessarily have to correlate and may be inversely related as there are no direct causal relationships between the latent variable and the formative indicators (Chin 1998).

The two major SEM approaches currently available for researchers are covariance-based SEM (CBSEM) and component-based approaches such as PLS. PLS was used as an analytical method to investigate the relationships between the constructs in the research model. PLS, as a variance-based SEM, was used since it is more appropriate than covariance-based SEM such as Linear Structural Relations (LISREL) and Analysis of Moment Structures (AMOS) for exploratory research (Petter et al. 2007). In addition, PLS approach has several strengths that made it appropriate for the current study, including the ability to test complex structural equation models with a large number of constructs, the ability to handle both reflective and formative constructs, and the fewer demands regarding sample size than other models (Chin 1998; Hair et al. 2014). The limitation of this study of a small sample size (160 observations) can be overcome by using PLS since the statistical power of PLS is always equal to or larger than that of covariance-based SEM (Hair et al. 2011). Moreover, the focus of covariance-based SEM is to estimate a set of model parameters such that the theoretical covariance matrix implied by the system of structural equations is as close as possible to the empirical covariance matrix observed within the estimation sample (Hair et al. 2014). This estimation requires the default assumption that the data are normally distributed.

However, PLS estimates the model parameters to maximise the variance explained for all endogenous constructs without assuming any distribution assumption and goodness-of-fit statistics (Vinzi et al. 2010).

7.3 Assessment of Measurement Model

The research model has one formative construct (i.e. agile EA), which consists of only one latent construct - agile EA (AEA). New measures were developed for the agile EA construct, based on the experts' feedback, as there are no existing measures in the literature, as discussed in Section 5.6.1. Agile EA was measured by how much EA is used in GDAD in aligning the project with business strategy and investment, involved in the solution architecture and used by distributed teams, and shared among distributed teams and team members.

Moreover, the research model has two reflective constructs: GDAD active communication and GDAD performance. GDAD active communication consists of two reflective latent constructs: communication efficiency (EFFIC) and communication effectiveness (EFFECT). The GDAD performance consists of four reflective latent constructs: on-time completion (TIME), on-budget completion (BUDGT), functionality (FUNC), and quality (QLTY).

EFFIC was measured by five items that were adapted from prior literature (Herbsleb & Mockus 2003; Piccoli et al. 2004), as discussed in Section 5.6.1. These items measure the extent to which the information among GDAD teams is shared quickly, easily, and with minimum cost. EFFECT was measured by four items that were adapted from prior literature (Herbsleb & Mockus 2003; Piccoli et al. 2004), as discussed in Section 5.6.1. These items measure the extent to which the shared information among GDAD teams is clear, detailed, and accurate.

Each of TIME and BUDGT were measured by two items adopted from prior literature (Jiang & Klein 2000; Lee & Xia 2010), as discussed in Section 5.6.1. These items measure the extent to which projects and tasks are completed according to schedule or according to pre-defined cost. According to Hair et al. (2014), a unidimensional two-

item construct is under-identified on its own. An over-identified confirmatory factor analysis (CFA) model may result when this construct is integrated into the overall measurement model. Although TIME and BUDGT consist of only two indicators for each, they are acceptable because they are sub-constructs of the higher construct (i.e. GDAD performance) in the overall measurement model.

FUNC was measured by three items adopted from prior literature (Mahaney & Lederer 2006), as discussed in Section 5.6.1. These items measure the extent to which GDAD project meets the functional and technical goals, and the customer requirements. QLTY was measured by three items adopted from the prior literature, also (Lee & Xia 2010), as discussed in Section 5.6.1. These items measure the extent to which GDAD project meets customer satisfaction and is able to solve customer problem.

Validating the measurement model is considered critical to ensure that a group of measurement items appropriately represent the concept (i.e. construct) under investigation (Straub et al. 2004). The measurement model validity tests were done by testing the reliability and validity of the measures used in the study (Hair et al. 2014). Validity and reliability tests were conducted to reflective constructs only by conducting EFA using SPSS analysis, as was discussed in Section 6.7 and for both reflective and formative constructs by conducting CFA using SmartPLS analysis, as will be discussed in this chapter.

Reflective and formative constructs should be treated differently. This is because, for formative constructs, different indicators are not expected to demonstrate internal consistency and correlations, unlike reflective constructs (Chin & Todd 1995). Instead, the relevance and level of contribution of each indicator were assessed by examining the absolute indicator weights. The content validity of the measures, the reflective measurement model evaluation, and the formative measurement model evaluation are discussed in Section 7.3.1, Section 7.3.2, and Section 7.3.3 respectively.

7.3.1 Content Validity

Content validity assesses the extent to which the indicators capture the major facets of the construct (Hair et al. 2014). Content validity refers to the issue of representation. That is, whether or not the instrumentation operates in a representative manner from all of the possible ways that could be used to measure the content of a given construct (Lewis et al. 2005; Straub et al. 2004). Some of the practical methods to measure content validity were identified through literature reviews and expert opinion (Straub et al. 2004). According to Lewis et al. (2005) guidelines, the measurement content validity was evaluated through several steps, as was discussed in Section 5.6.2.

7.3.2 Reflective Measurement Model Evaluation

According to Hair et al. (2014), the reflective measurement model was estimated by calculating four values: (1) internal consistency and composite reliability (CR), (2) individual indicator reliabilities, (3) convergent validity, and (4) discriminant validity.

7.3.2.1 Reliability

Firstly, to estimate the initial reliability to the measures, we have conducted the EFA by running the internal consistency test (as we did for the pilot study in Section 5.6.2.2) for the full-scale reflective measures. This test is conducted to increase the accuracy of measurement and ensure that the measures are indeed related to each other as they should be. Internal consistency was assessed by Cronbach's alpha, which should be greater than 0.7 to retain the item (Straub et al. 2004).

Cronbach's alpha was calculated for each construct separately. Appendix 7.1 illustrates the items, the Cronbach's alpha values of the constructs, and the Cronbach's alpha if item is deleted. All Cronbach's alpha values of all constructs were found above the cut-off value of 0.7, as shown in Table 7.1. Hence all items were retained and none was deleted. For most of the constructs and their respective items, the value of the relevant Cronbach's alpha would be decreased when deleting items from the scale. Therefore, the research reflective instrument remains at 19 variables from the six constructs (i.e. EFFIC, EFFECT, TIME, BUDGT, FUNC, and QLTY).

Table 7.1: Reliability statistics for reflective constructs

Latent Construct	Composite Reliability	Cronbach's Alpha
EFFIC	0.947	0.930
EFFECT	0.947	0.926
TIME	0.977	0.952
BUDGT	0.976	0.950
FUNC	0.889	0.813
QLTY	0.898	0.829

Secondly, the measurement model reliability was assessed using the CFA by testing the internal consistency reliability through a two phases approach: (1) composite reliability (CR) which is a measure of internal consistency reliability, and (2) individual indicator reliabilities which refers to the degree to which the measurement item is free of random errors and yields consistent and stable measures over time (Hair et al. 2014).

On the one hand, the reliability indexes of latent constructs were evaluated by CR, which considered acceptable if it is 0.70 or higher (Hair et al. 2014). All reflective constructs in our study achieved scores above the recommended value of 0.70 for composite reliability as shown in Table 7.1.

On the other hand, the indicator reliability of reflective items was assessed using the outer loadings of each item on its respective latent construct. The outer loadings of the reflective constructs should be above the recommended threshold of 0.708 and t-statistical significance value should be above than 1.96 (i.e. $P < 0.05$) [Hair et al. 2014]. The outer loadings of the reflective indicators were found all well above 0.708 as shown in Table 7.2. To measure the significance, the PLS-SEM relies on a nonparametric bootstrap procedure (Chin 1998). This technique uses re-sample sets in order to obtain the estimation for each parameter in the PLS model. Sub-samples are created with randomly drawn observations from the original set of data (with replacement), which are then used to estimate the PLS path model. This process is repeated until a large number of random subsamples have been created, typically about 5000 in order to obtain a stable result (Hair et al. 2014). The parameter's values (e.g., outer weights, outer loadings and path coefficients) estimated from the subsamples are used to derive standard errors for the parameters, and t-values are calculated to assess significance for

each parameter value (Hair et al. 2014). The current research conducted a "complete" bootstrapping with a sample size of 5,000 and a case size of 160.

Table 7.2: The cross-loading of reflective constructs and their items

Scale Items	EFFIC	EFFECT	TIME	BUDGT	FUNC	QLTY
EFFIC1	**0.883*****	0.423	0.631	0.440	0.495	0.458
EFFIC2	**0.883*****	0.393	0.594	0.516	0.489	0.496
EFFIC3	**0.931*****	0.474	0.594	0.473	0.456	0.506
EFFIC4	**0.907*****	0.416	0.578	0.464	0.381	0.444
EFFIC5	**0.815*****	0.506	0.598	0.510	0.385	0.415
EFFECT1	0.488	**0.897*****	0.367	0.438	0.601	0.562
EFFECT2	0.394	**0.890*****	0.351	0.498	0.621	0.635
EFFECT3	0.467	**0.908*****	0.333	0.381	0.547	0.660
EFFECT4	0.461	**0.922*****	0.399	0.430	0.574	0.546
TIME1	0.663	0.392	**0.977*****	0.799	0.407	0.405
TIME2	0.662	0.392	**0.977*****	0.804	0.433	0.417
BUDGT1	0.518	0.471	0.793	**0.975*****	0.494	0.489
BUDGT2	0.543	0.474	0.808	**0.976*****	0.469	0.427
FUNC1	0.483	0.599	0.415	0.472	**0.908*****	0.618
FUNC2	0.375	0.464	0.391	0.365	**0.785*****	0.525
FUNC3	0.418	0.592	0.295	0.420	**0.864*****	0.712
QLTY1	0.421	0.564	0.239	0.285	0.592	**0.825*****
QLTY2	0.416	0.567	0.460	0.508	0.654	**0.852*****
QLTY3	0.520	0.591	0.396	0.427	0.637	**0.913*****

Significance (p-value): *p < 0.10 (t>1.65), **p < 0.05 (t>1.96), ***p < 0.01 (t>2.58).

7.3.2.2 Convergent Validity

Convergent validity refers to the extent to which a measure correlates positively with alternate measures of the same construct (Straub et al. 2004). Convergent validity was assessed using EFA (i.e. Kaiser-Meyer-Olkin, Barlett's test, correlation), as discussed in Section 6.7 and using CFA by estimating the average variance extracted (AVE), as discussed below.

Unlike CB-SEM methods, PLS-SEM cannot test the underlying dimensionality of the data. In PLS, dimensionality should be assumed a priori on latent variables. However, unidimensionality, which states that a group of measures should represent only one construct, is a fundamental requirement for good psychometric measures (Straub et al.

2004). Therefore, EFA was conducted to address the unidimensionality condition through identifying the structure of the measurement model. As it was discussed in Section 6.7, the adequacy of the sample size and the undimensionality were met. Hence, we can proceed with our CFA analysis and examine the AVE values for reflective constructs. AVE should be at least 0.5 (Chin & Todd 1995) which indicates that the construct explains more than half the variance of its indicators. All AVE values were found well above the required minimum of 0.5 as shown in Table 7.3.

Table 7.3: AVE statistics for reflective constructs

Construct	AVE
EFFIC	0.783
EFFECT	0.817
TIME	0.955
BUDGT	0.952
FUNC	0.729
QLTY	0.747

7.3.2.3 Discriminant Validity

Finally, discriminant validity was assessed using two methods (Hair et al. 2014): (1) Fornell-Larcker criterion (Fornell & Larcker1981), and (2) examining indicators' cross-loadings. Discriminant validity refers to the extent to which a construct is truly distinct from other constructs by empirical standards (Straub et al. 2004).

Firstly, Fornell-Larcker criterion was used to examine the discriminant validity, which requires that the square root of each construct's AVE values should be greater than its highest correlation with any other construct. It is based on the idea that a construct shares more variance with its associated indicators than any other construct. This was confirmed in our findings (see Table 7.4), as shown by the numbers along the diagonal in bold font (square root of AVE) and the correlation with other latent constructs (off-diagonal elements).

Table 7.4: Correlation matrix for reflective constructs

Construct	EFFIC	EFFECT	TIME	BUDGET	FUNC	QLTY
EFFIC	**0.885**					
EFFECT	0.500	**0.904**				
TIME	0.678	0.401	**0.977**			
BUDGT	0.544	0.484	0.821	**0.976**		
FUNC	0.501	0.649	0.430	0.493	**0.854**	
QLTY	0.526	0.664	0.421	0.469	0.725	**0.864**

Secondly, discriminant validity was validated by examining indicators' cross-loadings. This test suggests that an indicator's outer loading on the intended (associated) construct should be greater than all of its loadings on other constructs. This was confirmed in our results as shown in Table 7.2.

Moreover, more recently, Henseler et al. (2015) reported that these approaches (i.e. Fornel-Larcker and cross-loadings) do not reliably detect the lack of discriminant validity in common research situations. The authors have proposed an alternative approach, based on the multi-trait multi-method (MTMM) matrix. This approach is called hetero-trait mono-trait (HTMT) ratio of correlations. HTMT ratio should be less than 0.9 between latent constructs (Gold et al. 2001). As Table 7.5 shows, the HTMT values between any two constructs were found less than 0.9. This supports the above results (i.e. Fornel-Larcker and cross-loadings) that discriminant validity was met. Overall, the reliability and convergent and discriminant validity (construct validity) were supported in the research model.

Table 7.5: HTMT ratio values for reflective constructs

Latent Construct	EFFIC	EFFECT	TIME	BUDGT	FUNC	QLTY
EFFIC						
EFFECT	0.540					
TIME	0.720	0.427				
BUDGT	0.578	0.515	0.863			
FUNC	0.572	0.745	0.489	0.559		
QLTY	0.596	0.759	0.476	0.531	0.885	

7.3.3 Formative Measurement Model Evaluation

The reliability and convergent and discriminant validity for the formative construct AEA were assessed using two methods (Hair et al. 2014): (1) collinearity which refers to the existence of high correlations between formative indicators, and (2) indicator validity. Firstly, collinearity was assessed by computing tolerance and VIF. Tolerance (1/VIF) refers to the amount of the variance of one formative indicator that is not explained by other indicators in the same construct. In PLS-SEM, a tolerance value of 0.2 or lower and a VIF value of 5 or higher indicate a potential collinearity problem (Hair et al. 2014). If the VIF value is greater than 5, the conflicting item should be removed as long as the overall content validity of the construct measures is not compromised. Other authors such as Diamantopoulus and Siguaw (2006) suggest that the criterion VIF for formative construct should be <3.3. The tolerance and VIF estimates for the measures of AEA are shown in Table 7.6. The results suggested that the reliability of AEA is supported and all indicators can be retained.

Table 7.6: VIF and tolerance statistics for formative indicators

Formative Indicator	Tolerance	VIF
AEA1	0.442	2.261
AEA2	0.491	2.036
AEA3	0.411	2.434
AEA4	0.416	2.402
AEA5	0.415	2.406
AEA6	0.480	2.083
AEA7	0.583	1.714

Finally, indicator validity was assessed by evaluating the significance and relevance of indicators' outer weights. Outer weight is the measure of the item's relative contribution to its assigned construct in the relationship from indicator to formative construct (Hair et al. 2014). Bootstrapping procedure (5000 samples) was used to calculate the weight's significance and t values for each indicator. The significant level is determined by the type of research. For exploratory research $p<0.1$ is acceptable (Hair et al. 2014). With seven uncorrelated indicators, the maximum possible outer weight is 0.378 ($1/\sqrt{7}$). Four indicator weights were significant as shown in Table 7.7. AEA2, and AEA6 weights values were not significant, but the weights were high (0.150 and 0.129, respectively),

while AEA 4 weight was only 0.002. However; as recommended by Hair et al. (2014), if the outer weights are not significant, the researcher can further evaluate the outer loading magnitude value which should be >0.5 and significant. If the loading is <0.5, but significant, the researcher might consider retaining or removing the items. If the outer loading is <0.5 and not significant, the researcher will need to decide whether to delete or retain the formative indicator for further analysis. The loadings for all items were above 0.5 and significant as shown in Table 7.7. The results suggested that AEA exhibited adequate convergent and discriminant validity. Accordingly, all items were attained (Hair et al., 2014; Peng & Lai 2012).

Table 7.7: Collinearity results for formative indicators

Formative Indicator	Outer Weights	t-Value	Sig.	Outer Loading	t-Value	Sig.
AEA1	0.308	2.266	0.024	0.815	12.246	0.000
AEA2	0.150	1.087	0.277	0.690	7.163	0.000
AEA3	0.244	2.309	0.021	0.833	17.008	0.000
AEA4	0.002	0.012	0.990	0.737	11.320	0.000
AEA5	0.267	1.830	0.067	0.775	13.299	0.000
AEA6	0.129	0.830	0.407	0.536	5.101	0.000
AEA7	0.381	3.447	0.001	0.797	13.951	0.000

Sig.=significance (p-value)

It is important to mention that eliminating formative indicators from the model should generally be the exception, as formative measurement theory requires that the measures fully capture the entire domain of a construct (Gotz et al. 2010; Hair et al. 2014). Moreover, the large number of indicators of the formative construct may cause such low weights of the indicators (Cenfetelli & Bassellier 2009). This occurs due to the competition between the indicators to explain the variance in formative construct (Cenfetelli & Bassellier 2009). For example, the maximum possible outer weight for AEA in this study is only 0.378 because seven indicators are forming the formative construct. In other words, omitting an indicator is equivalent to omitting a part of the construct, which leads to change in the meaning of the construct (Jarvis et al. 2003; MacKenzie et al. 2011). In addition, removing indicators with low weights removes the beta weight associated with these indicators (Petter et al. 2007). In fact, Henseler, Ringle and Sinkovics (2009) stated that conceptual justification for removing a formative

indicator is as important as statistical evidence, and all the items in this study were conceptually supported (i.e. expert recommendations).

According to Cenfetelli and Bassellier (2009), indicator weight only determines the relative contribution of the indicator to the formative construct. Although indicator weight is an important aspect for assessing formative indicators, it is not sufficient to remove the indicator. Instead of removing the indicator, Cenfetelli and Bassellier (2009) suggest that indicators with low weights should be tested against their absolute importance since low weights do not necessarily indicate the unimportance of indicators. The absolute importance can be tested by the indicators loadings, as discussed above, and their correlations with the formative construct (Cenfetelli & Bassellier 2009).

7.4 Assessment of Structural Model

The second phase of the model assessment is the validation of the structural model (Hair et al. 2014). In this study, it is argued that the use of agile EA can leverage and enhance GDAD active communication to enhance GDAD performance, and it also can directly enhance the GDAD performance. Moreover, active communication impact is directly and positively related to GDAD performance. As mentioned in the introduction, this study uses PLS-SEM analysis (Chin & Todd 1995; Hair et al. 2014) particularly with SmartPLS 3.0 (Ringle et al. 2015) to build the structural equation model and test the proposed hypotheses.

The structural model notion (general aim) is built on the fact that each link between any two constructs in the model represents a hypothesis to be examined (Barclay et al. 1995). The major emphasis when analysing the validity of a structural model is on four pillars: R^2 values of the model, testing the hypotheses between constructs by testing β significance, f^2 values, and Q^2 values (Chin 2010; Hair et al. 2014). Moreover, another analysis was conducted to examine the mediation role of communication efficiency and communication effectiveness. In addition, an extra analysis was conducted to test the impact of agile EA and GDAD active communication on overall GDAD performance. This was done by testing a modified PLS model where overall GDAD software

performance is modelled as a second-order construct that combines on-time completion, on-budget completion, software functionality, and software quality.

It is important to mention that R^2 and β in the PLS analysis are similar to the traditional regression approach regarding the results interpretation. However, PLS approach makes minimal distributional assumptions of sample size and normality (Gefen et al. 2011). Therefore, PLS analysis uses a nonparametric re-sampling procedure such as bootstrapping to estimate the significance (i.e. t statistics) of β (Chin 1998). Hair et al. (2014) recommended bootstrapping with 5000 resample. Therefore, the PLS algorithm and the bootstrapping procedure were run to assess the significance of the relationships, for all tests, with the following settings: mean replacement, no sign changes option, 160 cases, and 5,000 samples (Gefen et al. 2011; Hair et al. 2014).

7.4.1 Model Predictive Power (R^2)

R^2 value represents the amount of variance explained by the model of the endogenous constructs, which is an indicator to the predictive power capability of the model (Barclay et al. 1995; Hair et al. 2010). R^2 is computed as the squared correlation between a specific endogenous construct's actual and predicted values (Hair et al. 2014). The acceptable values of R^2 can be greater than 0.67, 0.33 and 0.19 and viewed to be substantial, medium or weak, respectively (Chin 1998).

The values of R^2 are shown in Figure 7.1. Agile EA explains 33.7 percent of the variance in communication efficiency and 31.7 percent of the variance in communication effectiveness. Agile EA, communication efficiency, and communication effectiveness collectively explain 45.6 percent of the variance in "on-time" completion, 37.2 percent in "on-budget" completion, 53 percent in software functionality, and 53.2 percent in software quality.

The R^2 results are above satisfactory, and thus, support the validity as well as utility of the structural model. The R^2 values of software functionality and quality are large (0.530 and 0.532, respectively). The R^2 values of communication efficiency, communication effectiveness, on-time completion, and on-budget completion are

medium according to Chin 1998's rule-of-thumb for R^2 classification. Since R^2 is a valuable tool in assessing the quality of a PLS model (Hair et al. 2014), our model can be described as being of above moderate quality.

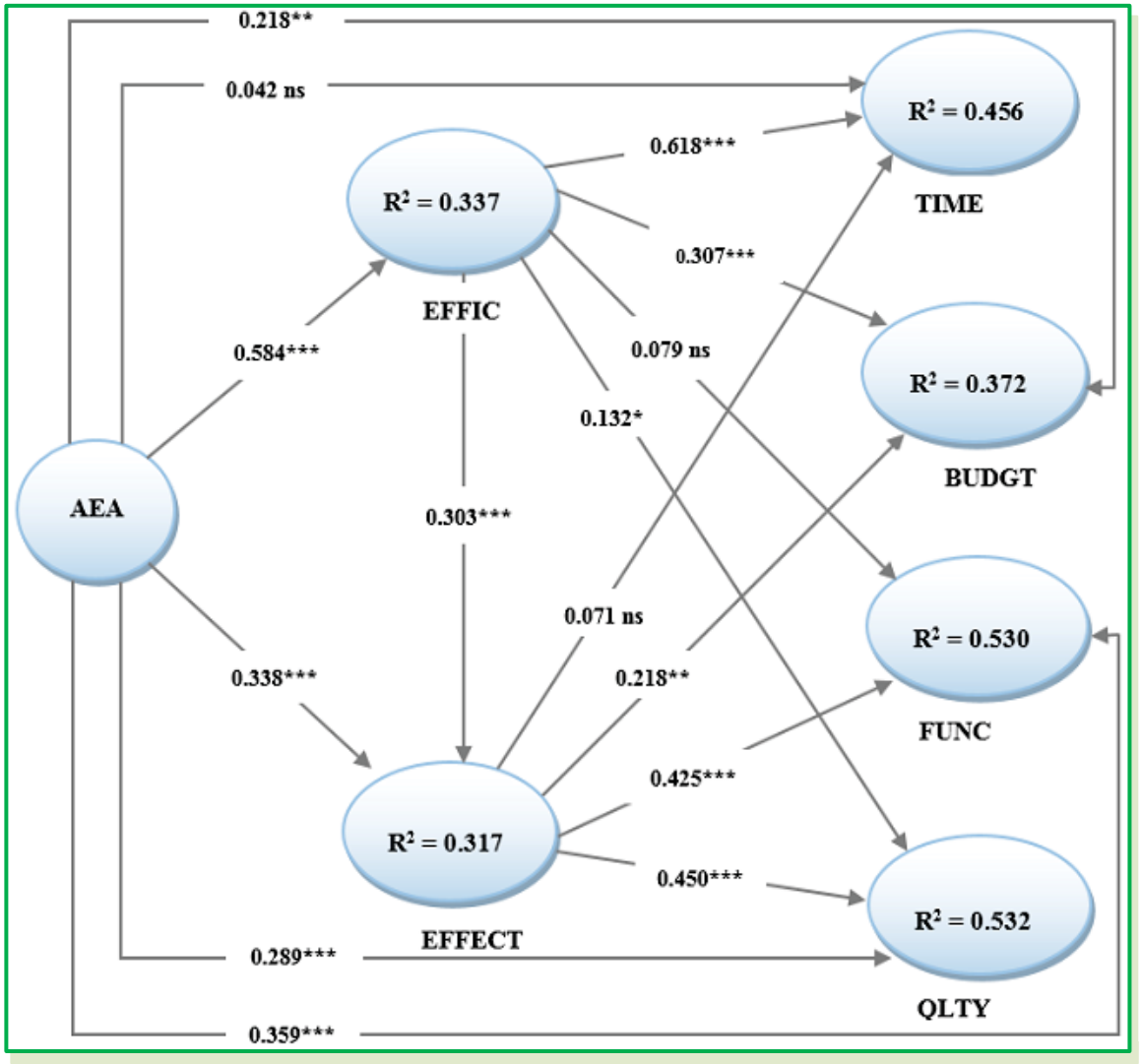


Figure 7.1. The partial least squares (PLS) results

7.4.2 Hypotheses Testing

β value represents the strength, direction, and significance of the relationship between two constructs. According to Cohen (1992), β is considered to be large, medium and small for values of greater than 0.37, 0.24 and 0.19, respectively. β should be significant and consistent with proposed directions following the standard suggested by Hair et al. (2011). The significance (p-values) and t-values are: ($t > 1.65$; $P < 0.1$), ($t > 1.96$; $P < 0.05$), or ($t > 2.58$, $P < 0.01$), which is equivalent to the 90, 95, or 99 per cent confidence

interval, respectively. A positive β indicates that the direction of the relationship is as hypothesised, whereas a negative β indicates the direction of the relationship is opposite to what is hypothesised. Table 7.8 shows the β values, the relationships directions, the t-statistics, and the p-values of the research hypotheses. Figure 7.1 illustrates these results (i.e. β values, t-values, and p-level). Most hypotheses were found significant. However, five hypotheses were found not supported. These results are discussed in the following sub-sections (Section 7.4.2.1, Section 7.4.2.2, Section 7.4.2.3, and Section 7.4.2.4).

Table 7.8: The partial least squares (PLS) results

Hypo.	Relationship	*β*-value	Sample Mean	Stand. Dev.	Stand. Error	t-value	Sig.	Remark
H1a	AEA→EFFIC	0.584	0.590	0.071	0.005	8.233	0.000	Supported
H1b	AEA→EFFECT	0.338	0.365	0.081	0.006	4.175	0.000	Supported
H1c	AEA→TIME	0.042	0.061	0.071	0.005	0.583	0.560	Not supported
H1d	AEA→BUDGT	0.218	0.237	0.091	0.007	2.407	0.016	Supported
H1f	AEA→FUNC	0.359	0.369	0.080	0.006	4.463	0.000	Supported
H1e	AEA→QLTY	0.289	0.298	0.072	0.005	4.016	0.000	Supported
H2	EFFIC → EFFECT	0.303	0.280	0.077	0.006	3.951	0.000	Negative support
H3a	EFFIC → TIME	0.618	0.612	0.077	0.006	8.049	0.000	Supported
H3b	EFFIC → BUDGT	0.307	0.302	0.082	0.006	3.748	0.000	Supported
H3c	EFFIC → FUNC	0.079	0.076	0.085	0.006	0.924	0.356	Not supported
H3d	EFFIC → QLTY	0.132	0.129	0.068	0.005	1.948	0.052	Supported
H4a	EFFECT → TIME	0.071	0.056	0.090	0.007	0.789	0.431	Not supported
H4b	EFFECT→BUDGT	0.218	0.200	0.092	0.007	2.367	0.018	Negative support
H4c	EFFECT → FUNC	0.425	0.412	0.060	0.004	7.066	0.000	Supported
H4d	EFFECT → QLTY	0.450	0.441	0.071	0.005	6.364	0.000	Supported

7.4.2.1 Effect of Agile on GDAD Active Communication (H1a, H1b)

This section answers RQ3. As shown in Table 7.8, using agile EA had a very strong positive impact ($\beta=0.584$, $p<0.001$) on the efficiency of GDAD communication. Thus, H1a was supported. Moreover, using agile EA was found to have a strong impact

(β=0.338, p<0.001) on the effectiveness of GDAD communication. Thus, H1b was supported.

7.4.2.2 Effect of Agile EA on GDAD Performance (H1c, H1d, H1e, H1f)

This section answers RQ4. As shown in Table 7.8, using agile EA had a very small positive impact (β=0.042, p>0.1) on the on-time completion of the GDAD project. Thus, H1c was not statically supported, and therefore, should be rejected. Moreover, medium positive impact was identified of agile EA on on-budget completion (β=0.218, p<0.05), which supports H1d. The impact of using agile EA on software functionality was found to be large (β=0.359, p<0.001), and was found large on software quality (β=0.289, p<0.001) too. Thus, both H1f and H1e were statistically strongly supported.

7.4.2.3 Relationship between Communication Efficiency and Communication Effectiveness (H2)

The relationship between communication efficiency and communication effectiveness was found large (β=0.303, p<0.001) but positive, which is opposite to what it was hypothesised (as a negative impact). Thus, H2 was not supported. The reverse direction between the two dimensions could be due to the positive relationship between the two dimensions or due to the items used in the questionnaire negatively formed.

7.4.2.4 Effect of GDAD Active Communication on GDAD Performance (H3, H4)

This section answers RQ5. Communication efficiency was found to have a very large positive impact on on-time completion (β=0.618, p<0.001), and a large positive impact on on-budget completion (β=0.307, p<0.001). Therefore, H3a and H3b were strongly statistically supported. Moreover, communication efficiency on software quality was found to have a small effect (β=0.132, p<0.1), and thus, H3d was statistically supported. However, the impact of communication efficiency on software functionality was found not significant (β=0.079, p>0.1). Therefore, H3c was not statistically supported.

Communication effectiveness, on the other hand, was found to have a very large effect on both software functionality (β=0.425, p<0.001) and software quality (β=0.450, p<0.001). Therefore, H4c and H4d were very strongly statistically supported. However,

the impact of communication effectiveness on "on-time" completion was found not significant (β=0.071, p>0.1) and positive, which is opposite to what it was hypothesised as a negative impact. Thus, H4a was not supported. In addition, the impact of communication effectiveness on "on-budget" completion was found significant (β=0.218, p<0.05) but positive, which is opposite to what it was hypothesised as a negative impact. Therefore, H4b was not supported.

7.4.3 Effect Size (f^2)

The effect size, Cohen's f^2 (Cohen 1988), is a measure to evaluate whether the omission of a specified exogenous construct has a substantive impact on the endogenous constructs (Hair et al. 2014). The effect size is calculated as:

$$f^2 = \frac{R^2\ included - R^2 excluded}{1 - R^2\ included}$$

Where; $R^2\ included$ and $R^2\ excluded$ are the R^2 values of the endogenous latent construct when a selected exogenous latent construct is included or excluded from the model. f^2 values of 0.02, 0.15, and 0.35, represent small, medium, and large effects respectively of the exogenous latent construct (Henseler et al. 2009).

Table 7.9: Effect size (f^2) values for latent exogenous constructs

Endogenous Latent Variable	f^2 of AEA	f^2 of EFFIC	f^2 of EFFECT
EFFIC	0.517		
EFFECT	0.112	0.090	
TIME	0.002	0.433	0.006
BUGT	0.046	0.098	0.052
FUNC	0.166	0.008	0.264
QLTY	0.108	0.023	0.297

As show in Table 7.9, f^2 values indicated a very large effect size for agile EA (0.517) on communication efficiency, a large effect size for communication efficiency (0.433) on 0n-time completion, and strong effect size for communication effectiveness (0.297) on quality. Also, a medium effect size was identified for agile EA (0.112, 0.166 and 0.108) on communication effectiveness, functionality and quality, respectively. In

addition, a medium effect size was identified for communication effectiveness (0.264) on functionality. Finally, all other effects sizes were found small.

7.4.4 Predictive Validity of the Model

In addition to evaluating the magnitude of the R^2 values as a criterion of predictive accuracy, Stone-Geisser's (Q^2) value (Stone 1974; Geisser 1974) can be examined as a criterion of predictive relevance of the model. Q^2 value is a measure of the cross-validated redundancy, which predicts the data points of indicators in reflective measurement models of endogenous constructs (Hair et al. 2014). Therefore, all endogenous constructs were assessed for indicator cross validation redundancy and construct cross validation redundancy. To assess the predictive relevance of the endogenous constructs, the blindfolding procedure (i.e. sample re-use technique, which omits a part of the data matrix, estimates the model parameters and predicts the omitted part using the estimates) was conducted using the default number of iteration (i.e. D=7) (Ringle et al. 2015). Q^2 value larger than zero indicates that the exogenous construct (agile EA) have predictive relevance for the endogenous constructs (active communication and performance) under consideration, whereas a Q^2 value smaller than zero represents a lack of predictive relevance (Hair et al. 2014). The Q^2 values of all latent constructs were significantly greater than 0, indicating the exogenous constructs' high predictive power (Table 7.10). Also, the Q^2 values for indicators were greater than 0 as shown in Appendix 7.2.

Table 7.10: Cross-validation redundancy (Q^2) values for reflective constructs

Total	SSO	SSE	1-SSE/SSO (Q^2 Total)	Relative Impact (q^2)
EFFIC	800.000	594.663	0.257	0.123
EFFECT	640.000	476.201	0.256	0.078
TIME	320.000	180.671	0.435	0.203
BUDGT	320.000	204.214	0.362	0.063
FUNC	480.000	300.772	0.373	0.178
QLTY	480.000	292.927	0.390	0.217

The predictive power of the structural model is the predictive relevance's relative impact (q^2). According to Henseler, Ringle and Sinkovics (2009), values of 0.35, 0.15, and 0.02 indicate large, medium, and small relative impact (q^2). The effect strength (q^2)

is calculated manually by running two models for each latent variable (i.e. a model includes and a model excludes that variable) using the following equation:

$$q^2 = \frac{Q^2\ included - Q^2 excluded}{1 - Q^2\ included}$$

Where; $Q^2_{\ included}$ and $Q^2_{\ excluded}$ are the Q^2 values of the endogenous latent variable for each D-th iteration. The values of q^2 of reflective factors are shown in (Table 7.10).

7.4.5 The Mediating Effect

To establish that an independent variable (X) affects a distal dependent variable (Y) through a mediating variable (M), as shown in Figure 7.2, Baron and Kenny (1986, p. 1176) recommend three tests: "A variable functions as a mediator when it meets the following conditions: (a) variations in levels of the independent variable significantly account for variations in the presumed mediator (i.e., Path *a*), (b) variations in the mediator significantly account for variations in the dependent variable (i.e., Path *b*), and (c) when Paths *a* and *b* are controlled, a previously significant relation between the independent and dependent variables is no longer significant, with the strongest demonstration of mediation occurring when Path *c* is zero".

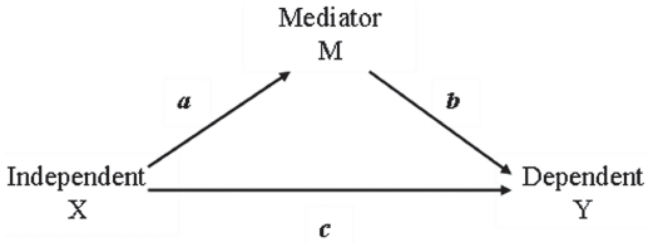


Figure 7.2. The mediation model

Although Baron and Kenny (1986) test (i.e. Sobel test) has been widely used, it is not a noble approach since it does not always discover the mediation effect if the data are not conforming to Baron and Kenny's criteria for establishing mediation such that the unexplained direct path can indicate an omitted mediator (Zhao et al. 2010). That is, it is possible to have significant positive correlations between X and M, X and Y, and M and Y and still have a significant negative indirect effect (Zhao et al. 2010). Sobel test "relies on distributional assumptions, which usually do not hold for the indirect effect. The multiplication of two normally distributed coefficients results in a non-normal

distribution of their product. Furthermore, the Sobel test requires unstandardised path coefficients as input for the test statistic and lacks statistical power, especially when applied to small sample sizes (Hair et al. 2014, p. 223).

When testing mediating effects, researchers should rather follow Preacher and Hayes (2004, 2008) and bootstrap the sampling distribution of the indirect effects, which works for simple and multiple mediator models (Hair et al. 2014; Zhao et al. 2010). According to Zhao et al. (2010), to establish mediation, all that matters is that the indirect effect is significant. The authors recommend establishing mediation by only one test: the bootstrap test of the indirect effect (*a* x *b*). The authors recommend checking first whether the direct effect *c* is significant, which tells what type of mediation or non-mediation we have, as follows:

- If (*a* x *b*) is significant but *c* is not, the mediation is indirect only,
- if (*a* x *b*) is not significant but c is, we have direct only non-mediation,
- if neither (*a* x *b*) nor *c* is significant, we have no effect non-mediation, and
- if both (*a* x *b*) and *c* are significant, determine the sign of (*a* x *b* x *c*) by multiplying the three coefficients, or by multiplying *c* by the mean value of (*a x b*) from the bootstrap output. If (*a* x *b* x *c*) is positive, it is complementary mediation; if (*a* x *b* x *c*) is negative, it is competitive mediation.

Hair et al. (2014) also proposed a detailed assessment for PLS mediation analysis. Sometimes, PLS path model without the mediator variable, a positive direct effect would become smaller after the inclusion of the mediator variable, which means that the mediator variable absorbs effect (Hair et al. 2014). To determine the extent that the mediator absorbs the effect in the structural model we calculate the variance accounted for (VAF) such that the indirect (*a x b*) should be significant. The formula to determine if VAF causes an indirect effect to total effect is: [(*a x b*)/ (*a x b + c*)], where all values of a, b, and c are the absolute values (Hair et al. 2014). Therefore, the extent to which the variance of the dependent variable is directly explained by the independent variable and how much of the target construct's variance is explained by the direct relationship via the mediator variable can be determined. If (0.2<= VAF<= 0.8), the mediation is

partial. If the VAF effect is larger than 0.8, the mediation is full. If the indirect effect is significant but does not absorb any of the exogenous latent variable's effect on the endogenous variable, the VAF<0.2, which means that almost no mediation takes place (Hair et al. 2014).

Table 7.11 reports the path coefficient (effect) between constructs. All indirect effects of agile EA on communication effectiveness and performance aspects (on-time, on-budget, functionality, and quality) were found significant. Also, the indirect effects of communication efficiency on on-budget, functionality and quality were significant, and insignificant on on-time completion.

Table 7.11: Effects results for the model

Relationship	Direct Effects	Direct Sig.	Indirect Effects	Indirect Sig.	Total Effects (direct + indirect)	Total Sig.
AEA→EFFIC	0.584	0.000			0.584	0.000
AEA→EFFECT	0.338	0.000	0.177	0.001	0.515	0.000
AEA→TIME	0.042	0.560	0.397	0.000	0.439	0.000
AEA→BUDGT	0.218	0.016	0.292	0.000	0.510	0.000
AEA→FUNC	0.358	0.000	0.265	0.000	0.623	0.000
AEA→QLTY	0.288	0.000	0.309	0.000	0.597	0.000
EFFIC → EFFECT	0.303	0.000			0.303	0.000
EFFIC → TIME	0.619	0.000	0.021	0.410	0.640	0.000
EFFIC → BUDGT	0.307	0.000	0.066	0.038	0.373	0.000
EFFIC → FUNC	0.078	0.356	0.129	0.002	0.207	0.036
EFFIC → QLTY	0.133	0.052	0.136	0.001	0.269	0.001
EFFECT → TIME	0.071				0.071	0.431
EFFECT → BUDGT	0.218				0.218	0.018
EFFECT → FUNC	0.425				0.425	0.000
EFFECT → QLTY	0.450				0.450	0.000

According to the above recommendation (Hair et al. 2014; Zhao et al. 2010), all the relationships which include significant indirect effect have some sort of mediation. Table 7.12 summarises them with the type of mediation. Full mediation was found between agile EA and on-time completion (communication efficiency and communication effectiveness mediated this relationship) with VAF value of 0.885, which means that 88.5 percent of the variance of this relationship is explained by communication efficiency and communication effectiveness. Also, full mediation was

found between communication efficiency and functionality (mediated by communication effectiveness), with VAF value of 0.618. The relationship between agile EA and communication effectiveness was partially mediated by communication efficiency such that 34.3 percent of the variance is explained by communication efficiency. No mediation was found between communication efficiency and on-time completion (VAF only 3.3 %). In addition, the relationships between communication efficiency and on-budget completion (VAF=17.7%) and quality (VAE=50.7%) were found partially mediated by communication effectiveness.

Table 7.12: Mediation effect results for the model

Relationship	Direct Effects	Indirect Effects	Total Effects	VAF	Mediation Type
AEA→EFFECT (mediated by EFFIC)	0.338***	0.177***	0.515***	0.343[6]	Complimentary mediation (partial mediation)
AEA→TIME (mediated by EFFIC and EFFECT)	0.042 ns	0.397***	0.439***	0.885	Indirect mediation only (full mediation)
AEA→BUDGT (mediated by EFFIC and EFFECT)	0.218**	0.292***	0.510***	0.340	Complimentary mediation
AEA→FUNC (mediated by EFFIC and EFFECT)	0.358***	0.265***	0.623***	0.081	Complimentary mediation
AEA→QLTY (mediated by EFFIC and EFFECT)	0.288***	0.309***	0.597***	0.200	Complimentary mediation
EFFIC → TIME (mediated by EFFECT)	0.619***	0.021 ns	0.640***	0.033	No mediation, only direct relationship
EFFIC → BUDGT (mediated by EFFECT)	0.307***	0.066**	0.373***	0.177	Complimentary mediation
EFFIC → FUNC (mediated by EFFECT)	0.078 ns	0.129***	0.207**	0.618	Indirect mediation only (full mediation)
EFFIC → QLTY (mediated by EFFECT)	0.133*	0.136***	0.269***	0.507	Complimentary mediation

[6] For example, VAF (AEA→EFFECT) = [β (AEA→EFFIC) * β (EFFIC→EFFECT)]/ [β (AEA→EFFIC) * β (EFFIC→EFFECT) + β (AEA→EFFECT)] = [(0.584*0.303) / (0.584*0.303 + 0.338)] = 0.177.0.515 = 0.343. All other values of VAF were calculated in the same way except for the values of VAF of AEA→ (TIME, BUDGT, FUNC, and QLTY), where these values estimated using the values of VAF in Table 7.13. For example, the VAF (AEA→TIME) = [VAF (AEA*EFFIC) * VAF (AEA*EFFECT)] / [{VAF (AEA*EFFIC) * VAF (AEA*EFFECT)} + β (AEA→TIME)] = (0.895 * 0.363) / (0.895*0.363 + 0.042) = 0.885.

Since the relationships between agile and the performance constructs were mediated by both communication efficiency and communication effectiveness, extra analysis was required to identify which communication dimension was responsible for this mediation role. We tested the indirect effect of each of the communication dimensions (efficiency and effectiveness) with agile EA to determine the mediator. The bootstrapping procedure was run with the following settings: mean replacement, no sign changes option, 160 cases, and 5,000 samples (Hair et al. 2014). We achieved the mediation effects (i.e. AEA*EFFIC and AEA*EFFECT) on all performance aspects (using the interaction term effect test in SmartPLS 3). We followed the above recommendation (Hair et al. 2014; Zhao et al. 2010) and included only the significant indirect effect. The effects of communication efficiency and communication effectiveness on the relationships between agile EA and performance aspects are shown in Table 7.13.

Table 7.13: Mediation effects for communication efficiency and communication effectiveness

Relationship	Effect β value	t- value	Sig.	VAF	Mediation Type
AEA*EFFIC → TIME	0.151	2.537	0.011	0.895	Indirect mediation only (full mediation)
AEA*EFFIC → BUDGT	0.255	3.519	0.000	0.450	Complimentary mediation (partial mediation)
AEA*EFFIC → FUNC	0.021	0.208	0.835	0.113	No mediation (very small mediation)
AEA*EFFIC → QLTY	0.107	1.633	0.103	0.210	No mediation (very small mediation)
AEA*EFFECT → TIME	-0.101	1.083	0.279	0.363	No mediation (very small mediation)
AEA*EFFECT → BUDGT	-0.108	1.216	0.224	0.250	No mediation (very small mediation)
AEA*EFFECT → FUNC	0.055	0.626	0.531	0.285	No mediation (very small mediation)
AEA*EFFECT → QLTY	0.143	1.793	0.072	0.344	Complimentary mediation (partial mediation)

As Table 7.13 shows, the relationship between agile EA and on-time completion was fully mediated by communication efficiency (explained 89.5 percent of this relationship); in fact, communication effectiveness insignificantly and negatively

impacts this relationship (β=-0.101). Therefore, the mediation effect is majorly identified by communication efficiency. Moreover, the relationship between agile EA and on-budget completion was partially mediated by communication efficiency (explained 45 percent of this relationship), while communication effectiveness insignificantly and negatively impacts this relationship (β=-0.101). Again, the mediation effect is majorly identified by communication efficiency. In addition, the relationship between agile EA and functionality was not found to be significantly mediated by either communication efficiency or communication effectiveness. So, it can be concluded that the mediation in this relationship is very small although collectively was significant as shown in Table 7.12. Therefore, we can conclude that none of communication efficiency or communication effectiveness majorly mediated this relationship. On the other hand, the relationship between agile EA and quality was found majorly mediated by communication effectiveness (explained 34.4 percent of this relationship) although the effect of communication efficiency was almost significant. Therefore, we can conclude that the major player in this relationship's mediation is the communication effectiveness.

7.5 Second-Order PLS Model

Higher-order modelling "involves summarising the lower components into a single multidimensional higher-order construct. This modelling approach leads to more theoretical parsimony and reduces model complexity" (Hair et al. 2014, p.40). Therefore, we tested a modified second-order PLS model to examine the effects of agile EA, communication efficiency, and communication effectiveness on overall GDAD software performance by combining all four performance measures. The tests of validity and reliability for a second factor model follow the same process used to examine the validity of the first order factor (Chin 2010).

The most suitable method for a reflective-formative model with a formative endogenous construct is a two-stage approach (Becker et al. 2012). Therefore, the two-stage approach was conducted such that the first-order latent constructs are considered as the indicators for the second-order latent construct (i.e. the latent construct scores of the four first-order constructs were extracted and used as indicators for the second-order

construct PERF). As shown in Figure 7.3, the second-order software performance (PERF) construct is a formative construct that has four indicators (the linear composites from the indicators used to measure each of the four aspects) representing the four aspects of software performance (on-time completion, on-budget completion, functionality, and quality). The t-values of the path coefficients between the second-order construct PERF and its first order factors were all greater than the minimum requirement of 1.96, indicating that the measurement model of the second order constructs yielded significant path coefficients.

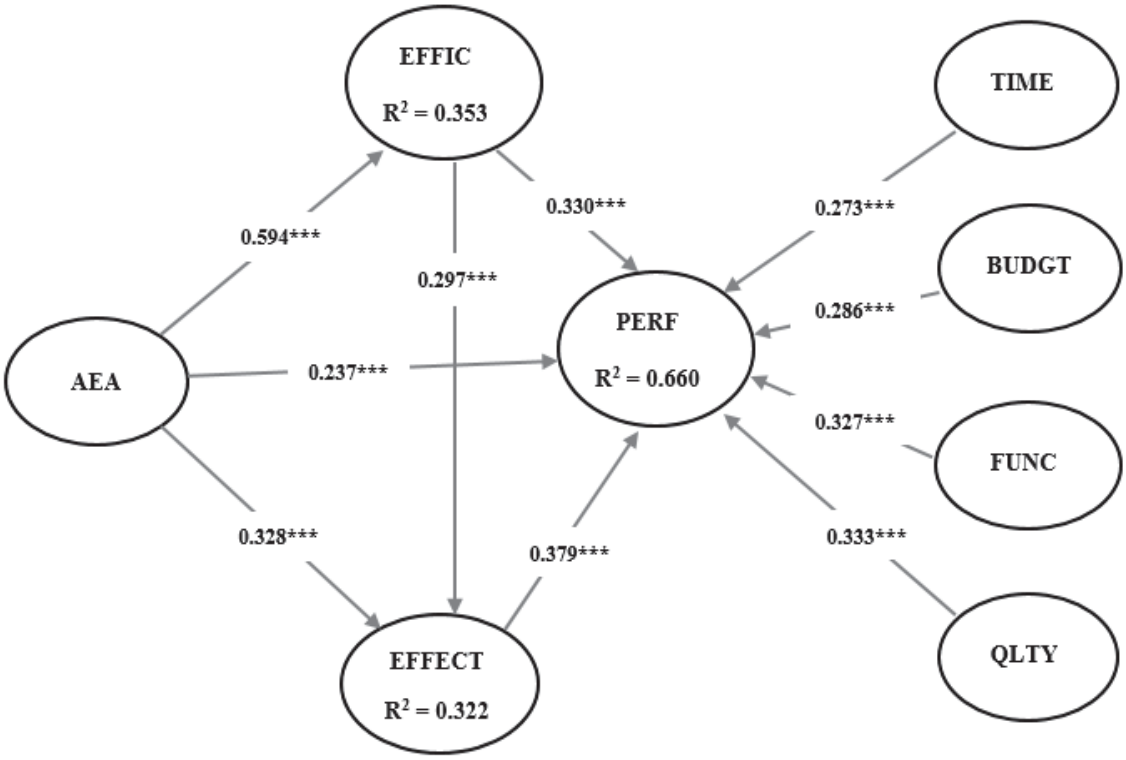


Figure 7.3. Results of the second-order PLS model

As shown in Figure 7.3, agile EA, communication efficiency, and communication effectiveness collectively explain 66 percent of the variance in software project performance (i.e. performance is largely explained by the second-order formative model), which supported the predictive validity of our model (Henseler et al. 2012). Agile EA explains 35.3 percent of the variance in communication efficiency (almost the same as the first-order model). Moreover, agile EA explains 32.2 percent of the variance in communication effectiveness (almost same as first order model). All paths between

software performance construct and agile EA, communication efficiency, communication effectiveness, on-time completion, on-budget completion, software functionality and software quality are significant. The coefficients for other paths (i.e. between agile EA and communication efficiency and communication effectiveness) remain stable between the second-order and the first-order models, suggesting the robustness of the research model. These results confirm the external validity of the second-order measurement model.

To test the validity of the second-order construct (PERF), we tested for the indicator level validity for the discriminant validity using VIF and indicator validity (indicator weight and significance of weights). These results are presented in Table 7.14. The VIF results, which are above the recommended value of 0.1, show that the multicollinearity is not a problem for the second-order model. Moreover, all the indicators' weights have shown a significance level (i.e. above the recommended value of 0.05). Therefore, the second-order model completely exhibits indicator level validity.

Table 7.14: Collinearity and validity results for second-order model

Indicator	VIF	Tolerance	Indicator Weight	t-Value	Sig.
TIME	3.077	0.325	0.273	15.803	0.000
BUDGT	3.333	0.300	0.286	15.072	0.000
FUNC	2.258	0.443	0.327	18.272	0.000
QLTY	2.230	0.448	0.333	21.156	0.000

7.6 Goodness-of-Fit of the Model

The goodness-of-fit is a measure of how well the sample data fits the conceptual model under investigation (i.e. how well the constructs of the model approximately represent the real-world phenomenon) (Gefen et al. 2000). Unlike CB-SEM, PLS-SEM model uses the sample data to obtain parameters that best predict the endogenous constructs (Hair et al. 2014). This is opposed to estimate parameters that minimise the difference between the observed sample covariance matrix and the covariance matrix estimated by the model. Therefore, PLS-SEM does not have a standard goodness-of-fit statistic. Moreover, goodness-of-fit is considered conceptually inappropriate for models involving formative measurement (Henseler & Sarstedt 2013). In fact, prior efforts to

establishing a goodness-of-fit statistic of PLS-SEM have proven highly problematic (Henseler & Sarstedt 2013). Instead, the assessment of the model's quality is based on its ability to predict the endogenous constructs using R^2, Q^2, β, and f^2 values. However, goodness-of-fit statistic may be useful for a PLS multi-group analysis, when the PLS-SEM results of different data groups for the same PLS path model are compared (Henseler & Sarstedt 2013). The R^2 values of communication efficiency and communication effectiveness, which are postulated as the predictors of the high performance, were above 25 percent, which is a highly acceptable prediction level in exploratory and social research (Beck et al. 2006; Gaur & Gaur 2006). Also, f^2 and Q^2 show medium effect (as shown in Table 7.9 and Table 7.10), which are also highly accepted in exploratory researches (Henseler et al. 2009; Hair et al. 2014).

Recently, Henseler et al. (2015) introduced the standardised root mean square residual (SRMR) which can be used as an indicator to goodness-of-fit measure for PLS-SEM. SRMR technique is based on transforming both the sample covariance matrix and the predicted covariance matrix into correlation matrices. SRMR is defined as the difference between the observed correlation and the predicted correlation (Henseler et al. 2015). Thus, SRMR allows the assessment of the average magnitude of the discrepancies between observed and expected correlations as an absolute measure of (model) fit criterion (Henseler et al. 2015). A value of SRMR less than 0.10 or of 0.08 (in a more conservative view) is considered a good fit (Henseler et al. 2015). Table 7.15 shows that SRMR value is less than 0.08, which indicates that the goodness-of-fit has been met for the research model.

Table 7.15: SRMR value

	SRMR	Sample Mean	Stand. Dev.	t-Value	Sig.
Common Factor Model	0.060	0.074	0.008	7.169	0.000
Composite Model	0.048	0.063	0.008	6.328	0.000

7.7 Chapter Summary

This chapter discussed the quantitative analysis results of a survey conducted with team members of software organisations that employed agile EA driven GDAD to examine if using agile EA can enhance GDAD communication and GDAD performance, and if

GDAD active communication can enhance GDAD performance. The measurement model was validated through a rigorous process that involved a number of systematic tests (i.e. reliability, convergent validity and discriminant validity, and indicator's collinearity and reliability). The content validity was first discussed and was previously assessed in Ch.5. Since the research model includes both reflective and formative constructs, PLS-SEM was the most suitable analysis to test the research model, and different tests were conducted to validate the measurement model. Reflective measurement model was validated by assessing the internal consistency reliability, convergent validity testing, and discriminant validity testing. Formative measurement model was tested by examining indicators' collinearity (using VIF measure) and reliability (using outer weights and loadings). PLS-SEM was used to assess convergent validity and discriminant validity, composite reliability and cross-loading of reflective indicators, and indicators' collinearity and validity. The results showed that all these tests were met. The measurement model developed showed sufficient rigour and was satisfactory, so we proceeded to test the structural model and the hypotheses.

Moreover, in this chapter the hypotheses developed in Ch.4 were tested through the validation of the full SEM using PLS approach. The SEM result suggested a strong impact of agile EA on active communication dimensions (communication efficiency and communication effectiveness), and strong direct and indirect impacts on software performance aspects. While agile EA directly and strongly related to on-budget completion, functionality, and quality, it was indirectly (fully mediated by communication efficiency) related to on-time completion. Moreover, communication efficiency and communication effectiveness had differential impacts, as it was hypothesised, on software performance aspects. While on-time completion and on-budget completion were strongly driven by the efficiency of the GDAD communication, functionality and quality were strongly driven by the effectiveness of the GDAD communication. Overall, ten out of fifteen estimates were consistent with the hypotheses. Furthermore, the model showed moderated predictive power (R^2>0.33) and predictive relevance (Q^2 well above 0). Chapter 8 discusses the qualitative findings of this study. The chapter provides insights to the supported hypotheses and explanations to the non-supported hypotheses of the research model.

Chapter 8 Case Study Findings

In Ch.7, the analysis results of the research model, its constructs, and research hypotheses were discussed. Strong positive relationships were found between agile EA and GDAD active communication (communication efficiency and communication effectiveness), between agile EA and GDAD performance (software functionality, software quality, and on-budget completion). Moreover, strong positive relationship was found between communication efficiency and communication effectiveness, communication efficiency and GDAD performance (on-time completion and on-budget completion), and communication effectiveness and GDAD performance (software functionality, software quality, and on-budget completion). Moreover, small significant relationship was found between communication efficiency and software quality. However, the relationships between agile EA and on-time completion, communication efficiency and functionality, and communication effectiveness and on-time completion were found not significant. Having all these relationships identified, we can now proceed in further and deeper investigation of these relationships by conducting semi-structured interviews with the industry people from agile EA driven GDAD organisations.

8.1 Introduction[7]

As it was mentioned in Chapter 5, this study employed a combination of quantitative (survey) and qualitative (case studies) approach to answer the research questions. This combination helps in gaining better understanding of the research phenomenon by addressing limitations for both quantitative and qualitative approaches by providing the objectivity of the statistics and deeper understanding of the study context. This chapter discusses the qualitative data analysis of the relationships between agile EA, active

[7] The findings of this chapter were partially appeared in Alzoubi & Gill 2017b

communication, and software performance in GDAD environment. 10 post hoc case studies were conducted that helped to validate the PLS results, provide explanations for the unsupported hypotheses, and provide further insights on the complex relationships among the constructs and beyond the quantitative results.

Following the qualitative data analysis model provided by Miles and Huberman (1994), an iterative in-depth exploration has been carried out to the textual data collected by interviews and the comments parts in the survey using a non-automated content analysis. The analysis aims to reduce the amount of text into more meaningful focused themes or categories by collecting notes about any relationships or patterns and then displaying extracted data as common or shared themes. Throughout the chapter, each section reflects a specific pattern or category (i.e. one of the three constructs of the research model: agile EA, active communication, performance or any related issue) under which relevant points or opinions are presented as direct quotations from the interviewees for easy referencing and discussion purposes.

This chapter is organised as follows. Section 8.2 presents the description of the cases. Section 8.3 discusses the validity and reliability validation of the collected data. Section 8.4 discusses the rationale of using the content analysis approach to extract the main themes from interviews and comments parts of the survey. Section 8.5 discusses the analysis steps used to analyse the qualitative data collected. The findings of the analysis are discussed in Section 8.6, before summarising this chapter in Section 8.7.

8.2 Cases Description

The sample for this study was 10 post hoc case studies of GDAD projects. Purposeful sampling was used to select cases that employed agile EA in their projects. The choice of cases in this study was also opportunistic based on industry relationships initiated by the researcher and supervisors. To preserve anonymity, the project cases are coded as A, B, up to J. Moreover, the quotations which were collected from the comments parts of the survey are coded as R1, R2, and so no.

Table 8.1: Case description

Case	AEA (H/M/L)	EFFIC/ EFFECT (H/M/L)	TIME/ BUDGT/ QLTY/ FUNC (H/M/L)	No. of teams/ Team size	Teams' Distribution	Interview Method
A1: program manager	H	M/M	H/H/M/M	5/ 2: 3-10 3: 10-30	3 countries/ 3 different time-zone	Face-to-face
A2: enterprise architect	H	M/H	H/H/H/H	5/ 1: 3-10 3: 10-30	3 countries/ 3 different time-zone	Face-to-face
A3: Product Owner lead	M	L/M	H/M/M/M	5/ 1: 3-10 3: 10-30	3 countries/ 3 different time-zone	Face-to-face
B: Product Owner	M	L/M	M/M/M/M	3/ 2: 3-10 1: 30-40	2 countries/ 2 different time-zone	Face-to-face
C: integration manager	H	M/M	H/H/M/M	4/ 2: 5-10 2: 10-30	3 countries/ 3 different time-zone	Face-to-face
D: delivery manager	M	L/M	M/M/M/M	3/ 10-30	2 countries/ 2 different time-zone	Face-to-face
E: iteration manager	M	M/M	H/M/M/M	3/ 1: 0-20 2: 20-30	3 countries/ 2 different time-zone	Face-to-face
F: Scrum Master	L	L/M	L/L/M/L	2/ 3-10	1 country/ 2 time-zone	Face-to-face
G: enterprise architect	H	M/L	H/H/M/M	6/ 1: 3-9 2: 10-20 3: 20-40	4 countries/ 4 time-zone	Face-to-face
H: solution architect	L	M/M	M/H/M/M	3/ 1:6-15 2: 20-30	3 countries/ 3 time-zone	Face-to-face
I: developer	H	M/M	M/M/H/H	3/ 10-20	2 countries/ 2 time-zone	Skype
J: Scrum Master	H	L/M	M/L/H/H	3/ 1: 3-9 2: 10-20	3 countries/ 3 time-zone	Skype

As shown in Table 8.1, these cases showed various combinations of different degrees of using agile EA, communication efficiency, communication effectiveness, and project performance. Most cases were involved in projects that involved more than 2 teams, and

most teams involved more than 8 members (the team size of the interviewee was highlighted in the table). Most teams were distributed between 2 time-zone and 2 physical locations. We interviewed different roles in agile teams; program manager, integration manager, delivery manager, enterprise architect, iteration manager, solution architect, Product Owner, Scrum Master, and developer. Case A includes a program manager (A1), an enterprise architect (A2), and a stream architect (A3). According to Dingsøyr, Fægri & Itkonen (2014), all cases' organisations can be described as "medium scale" (i.e. 1-9 teams).

Moreover, the comment respondent's demographics collected from the survey are shown in Table 8.2. Most of the respondents were agile coaches. The team sizes varied from small size (1-3 members) to very large size (100+). Also, number of teams varied from 3 teams to 15+ teams. The teams were distributed in minimum 3 sites and 2 time-zones to 10 sites and 6 time-zones.

Table 8.2: Survey comment respondent's demographic

Respondent	Job Title	No. of Teams	Team Size	No. of Sites	Team Distribution
R1	Agile transformation leader	10-15	100+	10 sites	4 time-zone
R2	Agile coach	15+	100+	17 sites	6 time-zone
R3	Agile coach	3-5	1-3	3 sites	2 time-zone
R4	Agile coach	3-5	10-20	4 sites	2 time-zone
R5	Agile coach	6-10	4-10	6 sites	3 time- zone
R6	Agile coach	6-10	4-10	7 sites	4 time-zone
R7	Agile coach	15+	1-3	10 sites	2 time-zone
R8	Architect	3-5	21-30	3 sites	2 time-zone

8.3 Data Validity and Reliability

According to Yin (2009), construct validity refers to identifying valid operational measures for the concepts being studies. In this study, construct validity was enhanced by using multiple sources of evidence by conducting multiple case studies, which ensured triangulation of data processing and interpretation. This data followed a predefined interview protocol (see Appendix 5.3b) that ensured consistency in examining each case. Moreover, using Leximancer for drawing the concepts' map

coding data achieved a chain of evidence from raw data to conclusion. This was generally achieved by comparing the interviews data with the survey findings. Moreover, according to Sekaran (2003), the validity of information collected can be enhanced by comparing it with other sources of information. These techniques and procedures helped in consistency and correctness of the measures of the model's constructs.

According to Yin (2009), reliability refers to the ability of the study to repeat the same results. Two techniques were suggested by Yin (2009) to ensure reliability in case studies: using a case study protocol and developing a case study database. This was achieved by employing the interview protocol and saving all transcripts into word sheets.

To avoid the potential bias between the researcher and the interviewees, the researcher followed a number of systematic procedures while conducting the interviews. First, a research model was developed based on existing GDAD literature that discussed the communication issues. This model was the basis for generating interview questions, developing an interview protocol, collecting data, and guiding data analysis. To address the data inaccuracy and incompleteness, audio-recording and transcribing to all interviews were applied. In addition, audio-recordings were checked; transcribed and reported data were read and re-read before data analysis. This ensured the validity of data collected and findings reported.

8.4 Content Analysis Approach Used in the Study

Following the above recommendations (Miles & Huberman 1994), the textual data collected from the interviews and the comments parts of the survey were repeatedly explored. This helped in achieving a more focused form of data.

All transcripts were read multiple times to get more familiar with the contents. The main concepts and codes (according to the research model constructs and concepts) were recorded while reading. A textual paragraph or sentence was extracted from the text only if evidence of key words which was found related to the main concepts and

codes. The keywords found from the first reading were added to the main concepts and codes and used to extract the other data in the following interviews. To synthesise a concept from one interview with other concepts from other interviews under a main category, a pattern emerged or an explanation pertaining to a similar phenomenon should appear between concepts. All formed categories were read again to ensure they were different and to ensure the internal consistency (Marshall & Rossman 2014).

Literature argues that using full automated analysis approach using one of the available software packages such as Leximancer (Leximancer Company 2016) would improve the overall reliability of the results (Scott & Smith 2005). That is, using an automated approach would help in avoiding the potential errors that occur due to inaccurate human judgment and fatigue, which increase the reliability (Scott & Smith 2005). Moreover, using an automated approach decreases the potential bias in the analysis that may occur using the non-automated methods of analysis alone due to researcher's intentions (Scott & Smith 2005).

However, the main interest of the analysis is the validity of the findings. Using the automated content analysis technique make it quite difficult for the researcher to fully understand and recognise the broader meanings of the text and the communicative intent of the specific word usage with relation to its context (Gebauer et al. 2008). Moreover, tagging a sentence (or a group of sentences) as containing a concept only occurs if the accumulated evidence (the sum of the weights of the key words) is above a set threshold (Scott & Smith 2005). This may result in a low-profile concept since no attention will be paid by the researcher to that concept in text because no focus is given on that concept from the automated analysis. Therefore, using the non-automated approach of coding the textual data in this study is strongly supported since it will result in a high level of validity in the analysis results (Aloudat 2010).

Therefore, this research used the content analysis, as a non-automated approach, to code the textual data. However, to provide a level of validation to the non-automated analysis technique and to have more confidence in the reliability of the analysis results, the transcribed interviews and the comments from the survey were parsed together through an automated content analysis using Leximancer 4.5 (QSR International 2016).

8.5 Data Analysis

Qualitative analysis data includes: examining, categorising, tabulating, testing or recombining data (Yin 2009). According to Ritchie and Spencer (1994) model, qualitative analysis includes five stages: familiarisation, identifying main themes, coding, charting, and synthesis. As mentioned in the introduction and in Ch.5, the data analysis method used in this study followed the interactive model of analysis (Miles & Huberman 1994), as shown in Figure 8.1. Miles and Huberman model includes four iterative stages: data collection, data reduction, data display, and drawing conclusions. These models helped us to perform the content analysis of data extracted from both interviews and survey's comments parts. The stages of analysing the qualitative data in this study are discussed in the following sections.

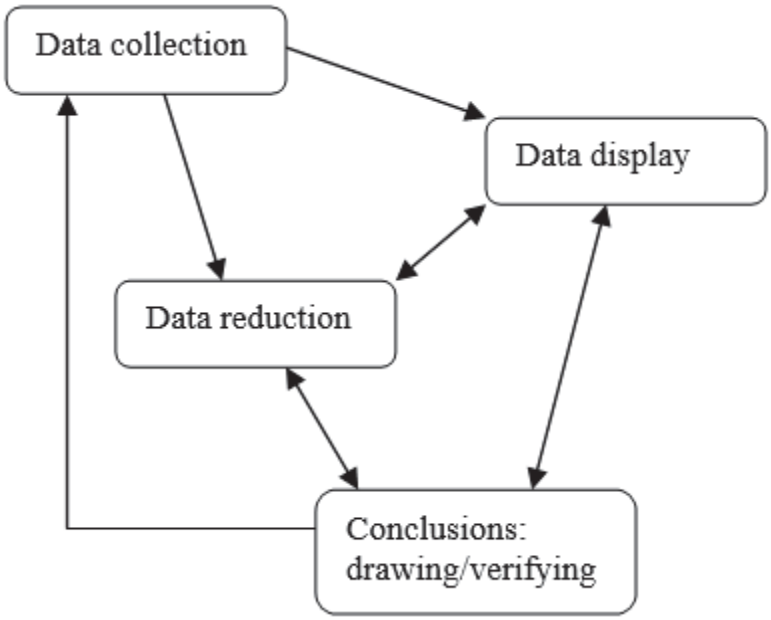


Figure 8.1. Data analysis interactive model (adopted from Miles & Huberman 1994)

8.5.1 Data Preparation and Familiarisation

After collecting data, as it was discussed in Section 5.7, data should go through preparation and familiarisation process to make it ready for analysis. Firstly, all audio records were transcribed into separate Windows Word sheets. Moreover, the handwritten notes and information that were taken by researcher during the interviews

were used as extra guidance and information for the analysis. In addition, all the comments from the survey (comments parts) were collected and saved in three separate sheets (i.e. agile EA, communication, and performance) according to each comments part in the survey. Secondly, the researcher took a close look at the data to become familiar with its richness, depth and diversity. Familiarisation was also achieved throughout the process of reviewing, organising, transcribing, and loading the data into Leximancer 4.5.

8.5.2 Data Reduction

Data reduction occurs continually throughout the analysis process (Miles & Huberman 1994). In other words, data reduction is a part of the analysis process and can't be separated. In order to transfer the data into usable form, preliminary coding was essential. This coding includes preparatory data reduction process by discarding some data such as the personal identifiable information and the appreciation sections of each interview. Then, reduction was conducted by identifying main themes. Here, the data was examined to identify key issues, concepts, and themes that appeared related to the research model constructs and relationships. Evidence was identified as disconfirmatory, confirmatory, or differing from the literature-based evidence in the model. This process was done iteratively within and across case data.

The coding scheme indexed the text that related to the presence of GDAD communication and communication challenges, the use of agile EA practices, communication related mechanisms and tools, and the impact of agile EA and communication (efficiency and effectiveness) practices on GDAD project performance (time or schedule, cost or budget, functionality or scope, and quality). Systematic coding allows seeing the logical link between the theoretical research model and the codes, and provides a means to avoid bias and validate interpretations (Dube & Pare 2003; Miles & Huberman 1994). The coded data was then grouped into related categories and/or linked for thematic examination.

8.5.3 Data Display

Data display is concerned with organising, compressing, and assembling the data into a more readable format than the original format, where data are scattered everywhere (O'Donoghue & Punch 2013). Therefore, data display was done in all stages of the analysis. Data display also is concerned about the stage of analysis that has been achieved and the preparation for the next stage of analysis (O'Donoghue & Punch 2013). Data can be displayed in different forms such as charts and tables (O'Donoghue & Punch 2013). Each set of codes of the interviews was examined to identify the elements that were primary to the category. These codes were then mapped into tables and diagrams for comparison with the elements in the reference model.

8.5.4 Data Synthesis and Drawing Conclusions

This is the final stage where the data charts/maps were synthesised against the research model and the conclusions were drawn about the similarities and differences. The goal of this stage is to generate a coherent and meaningful picture of the data (O'Donoghue & Punch 2013). Drawing conclusions could happen concurrently with other stages in the analysis (Miles & Huberman 1994). This stage is the hardest part of the analysis since it includes developing and verifying hypotheses, drawing conclusions, and confirming the obtained findings (Miles & Huberman 1994).

8.6 Qualitative Findings

As mentioned before, the interviews data were collected and analysed in the lenses of the research model. Therefore, the following sub-sections (Section 8.6.2, Section 8.6.3, and Section 8.6.4) discuss the qualitative results according to research model constructs (i.e. agile EA, communication efficiency, communication effectiveness, on-time and on-budget completion, functionality, and quality) and relationships. However, these constructs and the relationships between them are discussed in greater details, especially with regards to agile EA and GDAD communication, as the focus of this research.

8.6.1 Leximancer Results

Leximancer was capable of determining the core themes in the textual data and providing additional insight into the patterns that were explored in the content analysis approach, which helped the researcher to ensure that the themes in both findings (content analysis and Leximancer analysis) were matched. Figure 8.2 shows the main themes and concepts in the interviews and the comments parts in the survey that were automatically extracted from the Leximancer tool. The larger is the theme, the greater is its impact on the research model.

To overcome some of the limitations of the automated approach (as discussed above), some procedural interventions were applied within Leximancer: (1) clean the auto-generated thesaurus list of words such as “singular vs. plural”, “merging synonyms”, or merging together of words into short phrases like “agile EA-approach”; (2) delete the unrelated terms that may have been used regularly such as “think”, “must”, or “believe”; (3) add words, that were important according to the researcher but have not been considered by the software, to the visual concept map from the auto-generated thesaurus; and (4) consider the level of granularity of the concept map that reflects the relationships between the major concepts in the network.

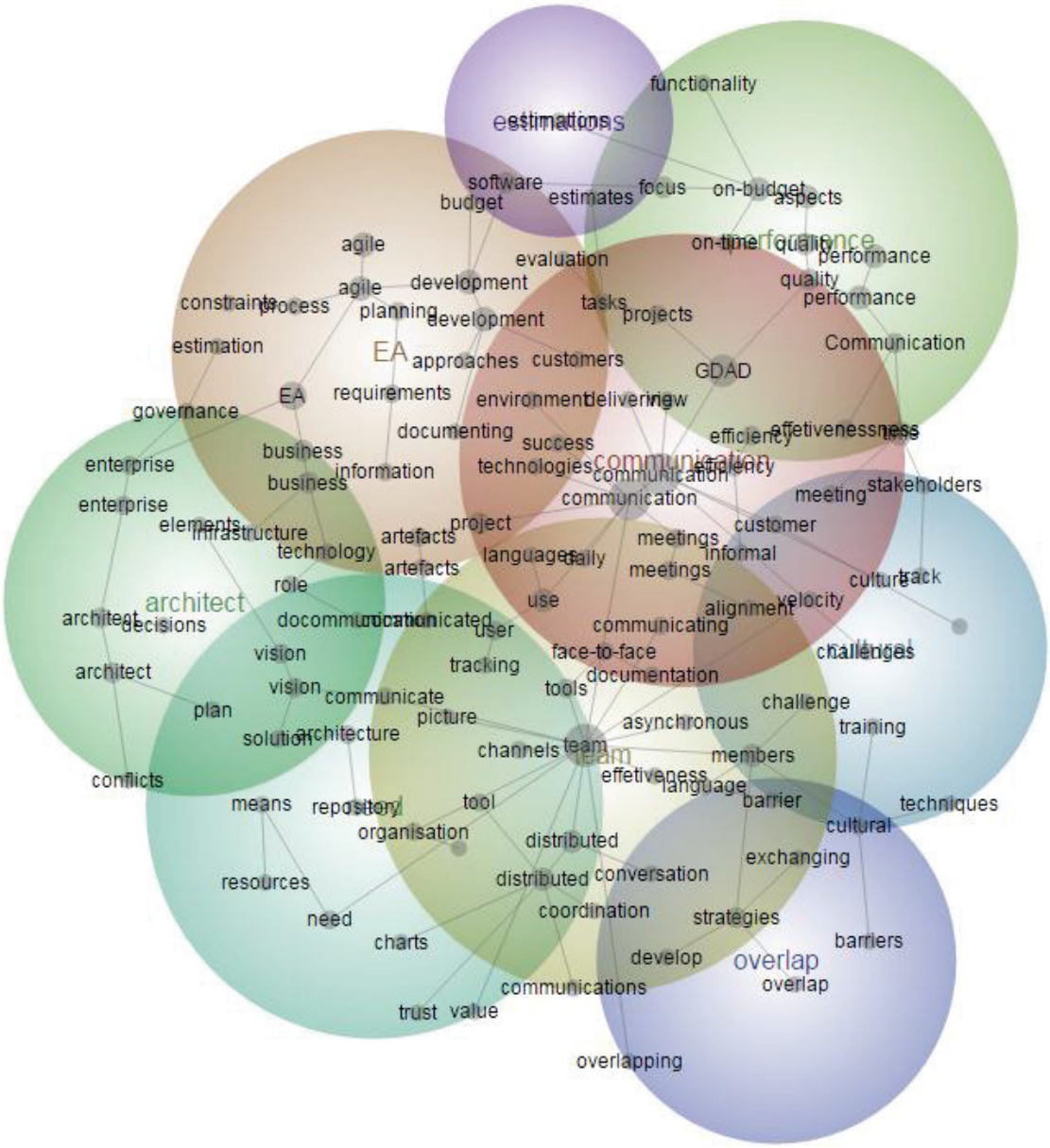


Figure 8.2. Leximancer concept map

8.6.2 Agile EA

Large organisations are appreciative of the EA role. As such, large GDAD organisations have started to recognise the importance of agile EA for their successful delivery. EA enables organisation to respond to changeable and disruptive requirements. More importantly, EA is an indicator of an organisation's internal and external strategic fit by tracing its goals into a business model. EA can become the means for any business transformation initiative which aims at transforming the current existing ad-hoc business activities into a new well engineered set of business capabilities. EA, as

described by Case G, "*not only enables organisations, especially big organisation, to survive in the currently non-stable economic environment, but also to grow and strategically win the competitive market"*.

In this section, all important themes that evolved during analysing the data regarding agile EA are discussed. In the following sub-sections, the definition of agile EA, the role of agile EA in an organisation, the agile EA vision, sharing agile EA vision, documenting agile EA, the role of architecture owner, the support of agile EA from portfolio level, and the conflicts between agile approaches and EA are discussed.

8.6.2.1 Definition of Agile EA

Different definitions are adopted by different GDAD organisations to agile EA. R1 states that: "*There are many flavours of EA or many people use it in different ways*". While some assume it as just as traditional EA that evolves during the development lifecycles, others believe that agile EA should have the abilities and principles of agile development. They define agile EA as a process that captures desired change and reflects the degree to which that change is accomplished. Agile EA should focus on the flexibility reaction to change; just as agile approaches do. R6 clearly identified the focus of agile EA as: "*Focus on the flexibility of change that is where I have found that EA teams need the most coaching. There is an iterative, incremental approach to building your EA, and that teams are key stakeholders in the refinement of the Product Backlog. It is the flexibility of agile methods that is the key attraction to my clients*". Case G also added that agile EA should "*respond to changing requirements*" (business, people, process, information and technology issues). Others define agile EA from a very high contextual view. They use the minimum level of EA details to get the work done; as it was formulated by R7 "*professionally, I always follow the EA mantra of 'I don't know', 'It depends' and 'Just enough detail' to do agile EA*".

Nevertheless, there is a general agreement among the interviewees that TOGAF can be used in agile, and thus, adopt TOGAF definition of agile EA. TOGAF defines "enterprise" as any collection of organisations that has a common set of goals". TOGAF also define EA as the practice for conducting enterprise analysis, design, planning, and

implementation such that fragmented legacy of processes are integrated into one environment that is responsive to change and supportive of the delivery of the business strategy. Case A2 stated that:

> *For me, agile EA follows the regular TOGAF definition. We use elements of TOGAF, but not the whole thing. It is kind of we use the TOGAF architectures of: business architecture, solution architecture, infrastructure architecture, and technology architecture.*

Agile EA is not a project "*that is ever completed*", as described by Case G. Agile EA continually evolves such that the future state becomes a new current state and updates agile EA content and plans accordingly. It's hard moving from a one-time event to an annual event and according to the budget planning, to an agile process that updates content of the leadership and governance activities of EA whenever strategic activities change the output of the architect activities. Case A2 explained that as: *"Essentially, what we do is the current state assessment, the final state, the previous state, and the gaps. The gaps then are used as opportunities. This is all in the initial base, basically"*.

8.6.2.2 Role of Agile EA

The purpose of agile EA is to connect and integrate all processes inside an enterprise in an optimising way. Agile can be used to denote both an entire enterprise that involves information, technologies, processes, and infrastructure or/and a specific domain within the enterprise. In both cases, agile EA crosses multiple systems and multiple functional groups within the enterprise. In addition, EA enables achieving the right balance between IT efficiency and business innovation such that individual business units can innovate safely. At the same time, EA ensures the integration of IT strategy with organisation's needs. Case A2 stated that:

> *It is very much aligning, so whenever business planning cycle happen, we feed in our IT requirements, and then get what business also wants from 1 year or 5 years' plan. Then we decide that we need to do this, we remediate this gap, or we build a capability. This is basically the EA*.

EA is the map that keeps all units on the same page. EA artefacts can be accessed any time and represent the whole common picture. Architects will help team members during the development lifecycle. This was identified by Case D as: "*EA build the whole picture, make sure it goes in the correct direction. If something goes wrong, we always can go back to the EA or the architects and ask for their help with that"*.

The use and acceptance of agile EA is determined by the social system in the development environment. Agile EA is people-oriented, which extends the scope of traditional EA that is a technical focus to view EA from a wider socio-political perspective. Case A3: stated that: "*there is a need to change from the dominating blueprint-focused EA to a relational-focused agile EA"*. As a part of this social system, the customer is the main focus of agile development. Hence, agile EA should pay the role of customer enough attention through communication, feedback, and so on. This was formulated by R2 as: "*EA covers a number of aspects but it should also define how GDAD plans interact with their customers, the mechanisms that would be used, and the manner in which the organisation would engage with customers during the development process*".

To realise the success of EA, value must be delivered. For most organisations, being able to particularly reveal quantifiable value delivery is critical. EA must constantly deliver provable value to maintain momentum and support from team members and the executive managements. Case C stated that:

> *Stakeholders need to see the value of our business work (side) so they appreciate and follow our plan. It is not just overhead work or putting work on. They need to see the benefit of that such as we did that and we are going to do that; so, all stay on the track.*

Surprisingly, EA role was assumed the same either the organisation using agile approaches in its development or traditional development. Since, as it was described by R8, "*EA is like city planning and agile is like building planning and EA is applied at*

strategic level and agile at execution level". This means that EA and development teams work separately in GDAD organisations. In other words, there is no difference between traditional EA and agile EA according to this opinion. Moreover, this opinion clearly assumes that an organisation makes its EA agile or not according to its approach. However, this opinion pays no attention to the social perspectives or to adaptation capability of agile EA comparing to the traditional EA. This opinion was also supported by Case G, who claimed that:

> *EA is nothing to do with your project, or how you actually do the project, or if you do agile or waterfall; it doesn't matter. Agile or waterfall is just ways to execute the project. If you follow TOGAF, agile development is located in the "Migration Planning" phase. So, your project is born in "Opportunities and Solutions" phase and when you execute the project you are migrating from current state to next state and that is where you do the execution*.

8.6.2.3 Sharing Agile EA Vision

The EA common vision should be created and marketed with simple and clear vocabulary to be easily understandable by GDAD teams. Uniform standards, checklists and examples should be provided to all teams. Such good tools are very helpful to share the EA vision with development teams (e.g., Wikis). These tools along with the central team repository (that has assistant artefacts such as use cases, story cards, tasks, events, questions and answers, tracking data, and personal blogs) provide all teams with EA vision. Case C clearly explained that as:

> *I think it is better to be more general when talk to off-shore teams and I don't go for that much details as I do with on-shore people. I will go for discussing the scope and time with off-shore teams and do not get them included in each single detail happened here, on-shore as this would be much better for business side.*

The core of EA is solidified by the on-site team who involves the most experienced people during the first iteration. This way; distributed developers can start up on a solid

architecture. However; off-shore teams (development) usually do not like EA, in general. This may be due to the EA constraints and requirements according to Case C who claimed that: *"Off-shore teams don't use EA artefacts as I use them myself as business manager. To be honest I think it scares them such as they think that we are not going to deliver till we meet all these requirements"*.

Case A3 stated that *''the major concern of using agile EA is how to communicate it and not the tools and frameworks, although these are necessary"*. Enterprise architect (s) and architects' teams usually underinvest the big effort of communicating EA. They know that it is critical for success and should be communicated throughout the enterprise units as it was suggested according to communication plan. The team of architects and enterprise architect need to regularly meet face-to-face. In addition, it is very helpful to use such communication tools that help in sharing EA. They also need to report to the executive level to seek their decision. Case A2 reported that: *"We have a chat group for me as enterprise architect and all architects. We also always meet together. We support our executives (CIO) such that we provide them with enough information to take decision. We use that group to report to our executives"*.

However, some cases reported that the execution does not match the plan, and indeed, the plan does not match the intent to communicate. Communication is assumed as the most time-consuming area, particularly in the case of GDAD where enterprise architect may need to travel to another continent to share and communicate EA vision. This was clearly stated by Case A2 who said: "*We communicate EA vision with all teams; local and distributed teams. Whenever any team needs clarity in some stage of the project, I travel and meet with architect of that team and discuss the state"*.

Organisational culture (organisation structure) reflects positively or negatively on communicating and sharing agile EA vision. While traditional hierarchy decreases the speed, and increases the communication efforts, the agile way was clearly positive in the speed of communication EA and in the relationships among architect's team. Case A2 reported that: *"We have different levels of architects; software, data, infrastructure, information, and technology. However, none of them report to me as enterprise architect. We all work as federated system. I, as enterprise architect, work*

collaboratively with other architects just like any architect. We all align with agile way".

After sharing and communicating EA vision with all architects (technical and business), EA vision is made available to all distributed teams online and should be shared among other teams (e.g., development teams, testers, analysts). According to the size of GDAD organisations, the architect (architecture owner) should communicate that vision with his/her team. Case G explained that as: *"At the beginning of the project, the vision is communicated among all architects and artefacts are hosted in online repository (document repository) so anyone needs it can go and have a look at it. In fact, it gives the direction to everyone. The digital leader or manager will go through that vision and details and share it and explain it to the other team's members*.

8.6.2.4 Documenting Agile EA

While agile prefers working software over comprehensive documentation, it does not cancel the importance of documentation. Agile EA must provide obvious utility to key stakeholders such as creating useful content that requires keen focus on the proposed stakeholders. Agile development would not work without the minimum level (at least) of documentation. In the real agile project, however, team members (e.g., developers) do not like to read documents. They prefer requirements and coding. This was reported by Case A2 as:

> *EA and architectures are all there documented. You can get as much as you like, but no one will read. In fact, as enterprise architect, when we deliver EA to the executives, they don't have enough time (2-5 minutes) to discuss EA. You need 2 minutes to summarise what you want. As far as all documents are available online for all architects to support their teams with whatever they need. All details are available, and they style it the way they want anytime they need it.*

In fact, using some sort of documentation is very helpful in tracking Sprint activities. For example, the communication tools and applications such as Wikies can be used to

track the key issue of the successes and failures in realising the desired future state through case studies, best practices, and fact-based research. Case E reported that: *"We use our communication tool to keep track of iteration and Sprint activities. This makes it easier in the sense that when you have that mad scramble when iteration tries to replace things together, but make it difficult at the beginning where we don't have clear requirements"*.

8.6.2.5 Architecture Owner Role

Large GDAD organisations may employ different architects such as solution architect[s], infrastructure architect, business architect, technology architect, and so on. Architect/ architecture owner role is important in large GDAD. He/she tries to resolve misunderstandings related to project's customers and business requirements between team members. The general role of enterprise architect, for example, is to communicate and share EA vision instead of outlining frameworks and blueprints. Case H explains his role as: *"As solution architect my job is to explain what should be done to different stakeholders who have poor knowledge about the architect and to convince them that following the architect will provide greater benefits"*.

Moreover, team members (e.g., development teams) return to the architecture owner always to declare or get a clear picture about the delivered software from technical and business point of view. This was clearly stated by Case G who said: *"Team can communicate with their representative architect anytime to discuss EA vision or if they face any question or problem"*.

Architecture owner was represented by the Scrum Master or Product Owner in some cases where the size of the team may be (3-9) members. Case F (Scrum Master) mentioned that he plays the role of architecture in his team of 7 members. He added "*I share and explain the EA vision regularly with my team as things going ahead, in stand-up meetings face-to-face. I also discuss the vision regularly with Product Owner, who represents the EA of the other team, using video conferences"*.

8.6.2.6 Support from Portfolio Level

GDAD organisations are moving away from traditional plan-driven approaches in order to decrease the cost and increase the benefits. Strategic planning with an agile mindset can provide a distinguishable shift in GDAD organisations that truly changes the game. This takes place if and only if EA undergoes continuous alignment of the business and technology teams. Case A1 clearly formulated that:

> *Planning of agile portfolio of large- scale organisation defines the strategy and decision on how much to invest is driven by value measurement as a project is incrementally released. Therefore, agile teams do work based on the strategically sense of the organisation on an ongoing and continuous basis. The cost and time are calculated based on resource allocation with a continuous consideration on what is paying off as expected*.

Although it's hard for some (who have been in the leadership position in traditional development hierarchy) to share it with the rest of the team and letting go of the control, agile approaches do not work with traditional hierarchical levels. Agile EA elements (vision) should be embedded in the daily iterations that all members know and share on producing solution architecture as the project evolves. *"Architects cannot accomplish EA aims by themselves; rather they, to a big extent count on others to do that. This usually requires an interpersonal approach based on friendliness, mutual understanding, and mutual trust"* (Case H).

Unlike traditional EA that requires proper governance, agile EA is a team product. In traditional plan-driven development, governance plans and decisions are defined without a link to enterprise architects, and vice versa. *"We don't have that hierarchy in our organisation. We all work as federated system"* (Case A2). While IT management works on IT governance and on partnering with people in the business, enterprise architects may be unconnectedly focused on EA processes and practices. This leads to a logical result, which is the *"conflicts and diverse roles and responsibilities and often misunderstanding result in significant wasted resources, overlaps, poor investment decisions, and miscommunications"*, as stated by Case G.

8.6.2.7 Conflicts between EA and Agile Development

Agile principles such as emergent design and lack of up-front planning were assumed as obstacles against the success of EA and solution architecture. This is because the enterprise architect or solution architect does not have enough time or up-front planning to accurately estimate the time, cost, and resources. This point of view was articulated by R4 who reported that: "*In agile development environment that has established EA and solution architecture functions, some issues come to surface due to the use of agile development itself, which are opposite to the activities of the architect, such as "emergent design" and a lack of sufficient information up-front that enable architect to accurately design the solution*".

Moreover, EA is applied at strategic level and agile applied as a method at execution level, according to interviewees and respondents. This means that enterprise architect or solution architect, for example have nothing to do with the development team. EA always helps in delivering successful projects (high performance) as it was mentioned by R3 who said "*EA is useful to teams, but it is their ability to make good decisions, collaborate, experiment, and function as a team. This makes the process of making decision comfortable and more valuable*". However, if the project fails, it means the execution level has failed to get the benefit of EA. In other words, it is the development team responsibility to get the full value of EA and make it in an agile way. This point of view was supported by R8 who stated that: "*Good EA leads to good successful projects but a good EA can't ensure success at agile execution level due to technical issue, people issues and other execution level failures. For example, at strategic level we can conclude that an enterprise EA planning can bring substantial cost benefits. So, at execution level if the EA project planning failed it's due to execution level difficulties. 50% of the projects don't go according to Plan*".

It is clear from the above two paragraphs and the discussion in Section 8.6.2.2 that the EA owner or architect is not part of agile development team. While this may be assumed as a fact in scaled-agile organisation where many architects may be included, it does not mean that EA artefacts or elements are not shared or communicated among distributed teams by a team member who can play the role of EA owner. In fact, we can conclude that these GDAD organisations may be using hybrid agile EA or do not use

agile EA (or what is assumed to be agile EA) at all. Rather they just follow the traditional EA planning, which may not result in full benefit in such a flexible and adaptive agile environment. The logical result from the agile development team will be the rejection or not "really following" EA plan. This means that the development team will not get the full benefit of EA, and the project will end up above budget or behind time line, which is a reasonable explanation to the high failure percent of GDAD projects.

8.6.3 GDAD Active Communication

The purpose of communication process in agile development is to communicate the project requirements, to define goals for the task definition, and deliver the solutions or projects. Tasks are written and described in the wall of the open office space using task board (based on the customers' stories). On-site team and distributed teams can see this either by personal attendance or using video conferences. When the iteration starts, the task board is used as a tool/media of communication and information sharing between distributed teams, and increases the visibility of the EA vision and common goals of the project. This allows continuous tracking to the product backlog, working tasks, and working processes. Moreover, the Sprint Backlog artefacts such as the excel sheet that includes project tasks and estimates for the next iteration can be a valuable practice for project daily status evaluation in the iteration retrospective meeting, as it was mentioned by R1 *"it is easier to discuss tasks and see the progress of my team using the story board"*.

In the following sub-sections, according to the interviewees and respondents' feedback, the GDAD communication challenges and the strategies and techniques used to mitigate them are discussed. While the focus of this chapter was on qualitative investigation of the findings of the quantitative results in Ch. 7, we noticed that there is a common theme among the interviewees that relates to communication challenges. So, we included these findings in the discussion in addition to discussing the two dimensions of active communication (communication efficiency and communication effectiveness).

8.6.3.1 Communication Challenges and Patterns

According to the cross-analysis of the case studies and comments parts of the survey, a number of challenges were identified to GDAD communication. The major sources of communication issues were: cultural differences, time difference, language barrier, communication technologies and tools, personal skills, and insufficient documentation. These challenges as well as some tips/techniques to overcome these challenges are discussed in the following sub-sections.

8.6.3.1.1 Culture Differences

Surprisingly, there was an agreement to a large extent among all cases that cultural difference was assumed as the biggest problem facing communication process between GDAD teams. This maybe because most of the cases interviewed have globally distributed teams (include multi-cultural teams). Case B formulated that:

> *The biggest problem we face is the culture differences. For example, the team in X is very hierarchical and they have their own signs of receiving the message such as "shaking heads", but sometimes it does not necessarily mean they got the point. While for the team in Y, they are very reluctant to ask questions during the meetings; however, they will come with many questions after meeting*.

The cultural differences require special skills of the on-site (main) team members (leaders) when communicating with different cultural teams. This, of course, adds more value to the expertise. Case C explained that:

> *Both teams in X and Y should be dealt with differently because of the difference in culture. In X, they are more focused on completing their tasks on time and scope. In Y, they take everything more personally (reflected on them) when we discuss an issue or the scope of the project. So you can't say for example to team members in Y you are doing bad job. You may say something like: it is better if we do it this way. You need to be very careful*

of what and how you report things also in our communication repository and tool.

Some tips and techniques were recommended by interviewees to overcome the cultural differences problem. These tips depend first of all on the experience (communication skills) of the team members, on the executives' strategies, and on the mutual respect to other members' culture when establishing GDAD communication. Case D summarises the communication skills as: *"You need to make sure that the message has been received as it was intended to be. If you rush and do not take your time, or you do not ask the question in the right way, you will have misunderstanding to your message. You need to take away the barriers between you and them so they talk. So, you need to ask the different type of questions such as "did you get it?", "say yes if you get it", "say no if no", and so on"*. Moreover, as team leader, you should ensure that all members are sharing and communicating with others. Case D added *"ask them and let them tell what they did yesterday, what they are doing today, and what they will do tomorrow. If they have a problem, ask them what I can do to resolve the problem"*.

From the executive management point of view, some strategies were recommended to mitigate the issue of different cultures among GDAD teams such as the training for team's members especially at the first beginning of their working in the organisation. This was made clear by Case C who reported that: *"Yes we do cultural training for all new members when the first joining the team"*. Another strategy was by getting the multi-cultural teams to tell what their cultures are. In other words, teams exchange the knowledge about differences in their cultures. This was formulated by Case A1 who mentioned that: *"We have the "buy-in", which is a cultural thing to make different cultural distributed teams tell us how things should work. You need to be flexible with other cultural teams. Agile is all about flexibility"*. The third strategy was by exchanging visits and rollover of the teams' members among distributed sites, which help in developing better understanding between members. Case E reported that: "*The new team members should come here, to Sydney and spend some time at the beginning of their joining our teams. Also, the leads or managers from Sydney do visits to different distributed teams, regularly"*.

The third tip that was recommended to overcome the cultural issue was by trust and respect for other cultures. This is common sense behaviour since if you show no respect to other cultures they will reply in the same way and show no respect to your culture too, which leads to a big failure or no communication at all between different cultural teams. The new team members should be introduced to everyone, which encourages them to be part of the common team spirit. Case E explained this pillar as: "*Our Company has a great culture such that everyone shows high respects to other teams or members wherever they are. For example, we do not do sessions if it is lunch time in X or Y where our partners base"*.

Another issue related to culture was the organisational culture that may negatively impact the communication and coordination among distributed teams, as it was mentioned by Case E; *"the other important issue of communication with off-shore teams is the organisation culture".* However, if the organisation adopts the culture that encourages the freedom, respect, and trust for other distributed teams, the communication may be increased. According to Case E, *"everyone in my organisation should show high respects to other members wherever they are. For example, we do not do sessions if it is holiday in our partners' countries"*.

8.6.3.1.2 Time Differences

The second major problem facing GDAD communication was the time differences (different time-zone) between GDAD teams. This challenge results in less synchronised communication and decreases the frequency of communication, in general between GDAD teams. According to Case E, "*the main issue with communication with off-shore teams is the "time-difference", which makes it very hard to make communication any time"*.

To decrease the impact of different time-zones, GDAD teams usually adjust their working hours to reach maximum overlap. Case D reported that: *"We do overlapping between teams".* Even though, sometimes it is hard to get all distributed teams in one meeting at the same time. In this case, as Case D added, "s*ometimes we just make sure that the majority attend at a time. However, teams' members can ask questions*

anytime". Moreover, it is not always even easy to find the right time to overlap between distributed teams. Case A1 complained that: *"It is always a challenge to find the right time for overlap since it is not sustainable to ask team members to work from 11am to 10pm, for example to collect all distributed teams in one-time meeting"*.

Another technique to decrease the time-zone differences effect is by employing communication tools that enable sharing and exchanging information. These tools have an extra advantage rather than face-to-face or synchronised video communication, for example, which is their ability to allow GDAD team members to have enough time to reveive and share the right message. This is very obvious since most GDAD teams speak another language rather than the main on-site team[s]. Case D formulated that: "*In addition to overlapping, we use group chat tool that enable sending questions and feedback anytime"*.

8.6.3.1.3 Language Barrier

The use of different languages rather than the main on-site team[s] language may limit communication between GDAD teams. This is true even if developers are skilful in common foreign business communication languages, such as English. Case E reported that: *"The other important issue with communicating with off-shore teams is the "language barrier" although everyone works for our company should speaks English"*. Language barrier issue was still noticeable even when multi-communication channels were used between distributed multi-lingual teams. This was reported by Case I who said: *"There were public forum and also private channels for communications, these were effective; however, most of the problems stemmed from the language barrier"*.

As it was mentioned in cultural issues, speaking slowly, informing team members about the topics of discussion in the meeting, and addressing the language issue as early as possible in the project are strategies that may reduce the impact of language barriers. In addition, some organisations run training sessions on using the main on-site team language such as English language. Case E formulated that: *"To overcome the language barrier, we have to run some training sessions in English for new members at the beginning when joining our teams"*.

8.6.3.1.4 Technologies and Tools

The fourth issue which faces GDAD communication is using the communication tools themselves. Although most practitioners and scholars recommended using these tools as substitutions to face-to-face communication, Case B clearly stated that: *"Nothing is going to be like face-to-face communication. However, you may reach that but that will take very long time. The communication tools should be used properly to give the maximum benefits"*.

That is, all distributed teams and team's members should be available to attend the synchronous communication or asynchronous tools as a substitution for face-to-face meetings. *"It is always about to use the right technology to support conversation and make sure the timing of those conversations is optimum"*, as reported by Case A1. Moreover, these tools are used to record and track the daily development progress. As a result, some members may not be ready or happy to do synchronous communication sometimes. Case E clearly reported that:

> *The fourth issue of communicating with off-shore teams would be the technology (tools), I think. We use teleconference ("TV Link" tool) to communicate between distributed teams either off-shore or near-shore. So, as a rule, every member should be connected to "TV Link" and keeps the web-cam on during the session time to share their screens with other members. Whatever we do (e.g., wall activities) here, in Sydney, are shared and accessible by all distributed teams, which keeps everyone aligned as much as possible.*

Communication tools can be used to support communication among GDAD teams. However, communication tools may negatively impact team communication if these tools are not tailored and systematically deployed to fit the purpose of GDAD. This will result in less efficient and effective GDAD communication. Therefore, GDAD organisations need to continuously check and update their communication tools, as it was mentioned by Case B: *"We do simulation to the tool twice a year to make sure it is*

effective as it supposed to be and to maintain the velocity of communication". Moreover, different tools can be good if one fails, the other can be used such as telephone, video conference systems, instant messaging, e-mail, virtual whiteboards and other technical means. The basic infrastructure used, however, should provide a high-speed data link. Case E suggested that: *"Technology is a big tip in establishing good GDAD communication such as what we use here "TV Link" and "chatting tool" that enables everyone to see all parts of the project parts is very helpful to communicate with distributed teams".*

8.6.3.1.5 Personal Skills

Personal skills were mentioned as another challenge for GDAD communication. Case E reported that: *"The other issue with off-shore communication would be the difference in teams' members' experience".* Most interviewees think that personal skills decrease with distance such that the most skilled members are those who work in the on-site team, followed by the those who are located in the near-shore or distributed in the same country as the main team (typically, inside Australia), and then by members in offshore locations (i.e. globally distributed in different countries). The worst of skills are those related to difficulties in communication and collaboration.

Therefore, the on-site main team is assumed to have the main resources in GDAD organisation, and generally cost more than other distributed teams. Case A3 claimed that: *"The main team which works as a hub is another tip to mitigate the skills differences. Sydney team works as a hub to all other distributed teams. All teams can watch and see what we do here in Sydney. This needs more effort from members in Sydney, but it enables all distributed teams to capture everything happened here"*.

In some scenarios employing domain experts located in distributed location is recommended to "*decrease the differences in personal skills among GDAD teams*" (Case G). This scenario may help in solving some of the communication and coordination problems that arise when remote teams are composed of less experienced personnel than the on-site main team. This enables more scalability as the on-site team is not as overwhelmed by remote site requests, and increase the trust between executive

management and the distributed teams as the distributed teams become more understanding of the business needs.

8.6.3.1.6 Documentation

Less documentation to the main artefacts was assumed as a challenge to GDAD communication by some interviewees. That is, face-to-face communication, in the co-located and local context, saves time and effort and reduces documentation, which increases the benefits for business and enhances customer satisfaction. However, in GDAD environment and due to lack of face-to-face communication, documentation of main artefacts and codes is important. In fact, these artefacts such as source code may represent the main communication medium between distributed sites. Case A2 reported that:

> *Definitely, otherwise how they will handover! You need at least minimal documentation, but not too much, and should be easily searchable, understandable, and accessible by all stakeholders*.

8.6.3.2 Communication Efficiency

As it was seen from all case studies, the on-site teams work in an open office place. This gives them the opportunity for more informal communication, which reflects positively on the efficiency of the communication during development iterations. This is because the open office environment is a way that provides a more efficient discussion and solution forum, on a daily basis, of the project progress among the development team. Case B reported that: *"All the team members sit in this big office. We have the open channel with the project leads all the way. It has been a great success. It is much faster than to solve any issue here than work with distributed teams"*.

In GDAD, it is much harder often to get the same efficiency of communication than with on-site (co-located) teams. As discussed above (see Section 8.6.3.1), many challenges face GDAD communication that decrease the efficiency of communication. As Case B reported in the above paragraph, communication efficiency within on-site teams is much faster than with GDAD teams due to less informal communication

available among GDAD teams' members. To achieve more efficient GDAD communication, interviewees recommended adopting both formal and informal communication using the available tools and strategies. While informal is preferable for agile, GDAD need more formal communication, especially with different spoken languages and multi-cultural teams. Case D formulated that as: *"We try our best to use our tools to make the communication as we do here in Sydney as face-to-face. It is hard to get to that point, but with time and more training we hope we can reach to the point"*. Moreover, the agile principles and practices such as meetings, testing and integration were seen as mechanisms of increasing the communication efficiency. Case F noticed that: *"Continuous integration is very helpful as you have the new build available for testing all the time"*. In addition, other interviewees reported that identifying resourses such as tasks and timelines among GDAD teams enhance the efficiency of GDAD communication. Case H described that as: *"The roles classification and responsibility, team hierarchy, and work-flow definition enhance communication efficiency"*.

Product Owner takes the role of real customers in large GDAD organisations. He/she contributes the customer input to the development team. He/she has the knowledge in the application domain and the ability and skills to interact equally well with technical teams and business team members. He/she carries out his/her communication with the customer through video conference systems, phones, emails, and face-to-face on a regular basis. In team's meetings (e.g., Sprint planning), Product Owner *"was the key person who regularly communicates the new requirements. This speed up communicating customer requirements among all teams"* (Case H).

8.6.3.3 Communication Effectiveness

Communication effectiveness was recognised more than communication efficiency among the interviewees. In general, they look at communication from the lenses of effectiveness, or they may define communication efficiency under (as a part) of communication effectiveness. The ineffective communication may lead to issues such as lack of cooperation between different distributed teams, and lack of understanding of the customer requirements. Case D defines the effectiveness of communication as: *"You need to make sure that the message has been received as it was intended to be"*.

As for communication efficiency (discussed above in Section 8.6.3.2), communication effectiveness is more important and easier to achieve within on-site teams than with GDAD teams due to the same reasons of less informal communication and many communication challenges. Open office environment was recommended as an effective strategy in motivating face-to-face discussion and facilitating design problem solving which was seen as a factor of facilitating the team's achievement of goals during a defined iteration. This also decreases the need for documentation due to the use of such practices. Case J commented: *"People, here (in Sydney), don't like to read much documentation. So, we use less documentation within on-site location. However, more documentation is still required with distributed teams"*.

In order to increase the effectiveness of the communication, all teams are forced to use only English during the meetings. Case E formulated that as: *"We have a policy that whenever we have a session, all conversation should be in English during the session, which increases the effectiveness of the communication"*. Culture issue was also raised by interviewees as another tip to increase the communication effectiveness among GDAD teams. In addition, special attention should be paid to ensuring that each participant is trusted as equal to all other team members, which enhances the effectiveness of communication. Case D reported that: *"You need to be aware to other cultures, understand other countries calendar such as their holidays, and treats everyone equally to make the correct communication"*.

According to interviewees, using available communication tools were recommended as a solution to increase the effectiveness of GDAD communication. R5 explained that: "A*cross all disciplines, I often ask distributed teams to maximise communications effectiveness just as if they were co-located. What would that look like? Open communications using Skype and removing barriers to delivering value to our customers are of paramount importance*". Moreover, GDAD communication among teams takes place using instant messaging, which is preferred over voice calls and face-to-face communication since it helps decrease the impact of communication challenges. Instant messaging is a good compromise between real-time and asynchronous communication. It also gives the distributed team members the chance to understand the message, especially if different language is used. In the meetings however, all GDAD

members are required to participate and ask questions. In order to ensure and increase the effectiveness of communication, Case D explained that: *"You need to ask the different type of questions such as "did you get it?", "say yes if you get it", "say no if no", and so on. You should be patient; take your time and give enough time for every member, communicate clearly, make sure all members understand what is the point or the message, and when you going through the meeting give the chance to talk. If you rush and do not take your time, or you do not ask the question in the right way, you will have misunderstanding to your message. You need to take away the barriers between you and them so they talk"*.

As for communication efficiency, agile principles such as Scrum meetings and Sprint planning are recommended as tips to increase the effectiveness of GDAD communication. For example, Sprint planning meetings and reviews have a positive impact on the feature–requirements dependency since the use of these practices provides a systematic way to share information on the features and requirements between all GDAD teams such as the testing teams, interface projects and customers. Case A1 stated: *"I meet with delivery lead and iteration manager every two days. We go to the wall of cards (Scrum of Scrum) in addition to other meetings with the other team members to keep the track and scope alignment"*.

8.6.4 GDAD Performance

The factors of high performance of the project, in general, from the business management point of view are usually on-time completion, on-budget completion, software functionality, and software quality. Other experts refer to the functionality as the scope of the project such as Case A1 who reported that: *"As program manager, my focus is about getting the team to focus on the scope that should deliver"*. However, it sounds to be very hard to achieve the designated levels of these aspects in GDAD environment. This was reported by Case A1 as: *"Those three aspects (time, budget, functionality) always have a constant battle, so time usually is against cost, and scope always fights against time and cost as well. This is harder in agile development than plan-driven, where vendors don't work agile. The situation is even harder with distributed agile"*.

Different organisations have different approaches of identifying their projects' performance aspects. The general approach among all interviewees is that at the first Sprint planning meeting, all stakeholders are involved in the meeting, all priorities are agreed, all tasks are defined and assigned to different teams, and all estimates are presented for each task. However, time and cost cannot be accurately estimated. Case A1 declared that: *"We always try to form up the cost up front in agile project, where quite often that the vendor cost would be much more than the estimated. The situation is even harder with distributed agile"*.

Although the quality should be delivered as agreed with customers and was generally reported very good from all interviewees, our interviews revealed that development problems related to quality are not uncommon. It was clearly identified that quality correlates negatively with on-time and on-budget completion. Late and over budget projects may result to achieve a good quality. Nonetheless, a trade-off on quality is often accepted to meet deadlines. This compromise was clearly identified by Case A3, who said: *"I think the on-time and on-budget are connected to the quality such that increasing quality will come on the expense of extra time and cost"*. Moreover, the trade-off can be seen even between on-time completion and on-budget completion, as it was identified by Case C who said: *"In general, I think there is always kind of trade-off between these aspects (time, budget, quality). So, you can't have all of them perfect in any specific project"*.

8.6.5 Relationship between Agile EA and Communication Efficiency

This section answers the first part of RQ3. Using agile EA vision decreases the issues of interactions or interfaces in GDAD environment, especially when many of features are supported by different people, since it is difficult to evaluate and implement interfaces between features and requirements. It also represents the joint hub between development teams and the management or business level. This vision helps GDAD teams to track their tasks and keep all teams on the same page. Agile EA artefacts can be represented by charts, diagrams, tables, and so on. Case C reported that:

> *I use this communication tool for all my reporting. We divide the vision in the releases. I use visual methods usually Pie charts to give a very visual representation to each stream, so I don't have to say it. They just look at the charts and can track very well. Then I do my velocity report, by which we scope the progress and find out the actions needed. I think this report helps the teams, especially the business people. I get them to identify the work, and then send it back to different teams*.

Agile EA vision involves the solution architecture. This vision is evolved and updated with new solutions, and saved in a repository. All teams can access this repository at any time. This was reported as a mechanism to increase the efficiency of the communication among GDAD teams. Case E formulated that: *"My Company has a repository that stores all solutions, which includes all these diagrams and charts, tables, etc. Usually, solution architect uses EA artefacts, and then shares it to his/ her team. Solution architecture helps a lot to speed up the conversation with distributed teams, especially off-shore teams"*.

Moreover, EA vision and repository represents a reference for all teams to get answers to their questions. The repository holds all the questions which were previously discussed, the comments, and the feedback. This was assumed as a significant GDAD communication efficiency tip among teams, especially from on-site leaders' point of view. Case A2 formulated that: *"Yes. It is very helpful because it gives them the directions we take to deliver our project. They have many questions and if EA was not there, answering these questions will be time consuming. So, EA vision makes the communication faster since they know all the connections in EA level"*.

8.6.6 Relationship between Agile EA and Communication Effectiveness

This section answers the second part of RQ3. As it was discussed above (Section 8.6.2) EA vision and artefacts can be used as a blueprint (EA artefacts specify a certain implementation or provide guidelines for an implementation on a higher level) and as a common language (i.e. EA artefacts can be used as a communication medium in many

situations, and existing EA artefacts can be used to guide the creation of the new ones). This vision was seen as a powerful driver to conversation among distributed teams. Case E reported that: *"I think putting everything on our chatting tool so all teams' members can see and print is very helpful. Moreover, we send our iteration reports which include charts, tables, and all other details to all distributed locations, which is really a powerful tool to help ride the conversation so they know what we are going to have. I believe charts, tables, and other artefacts are really big pieces of working distributed agile"*.

As this vision and these artefacts were helpful to communication efficiency, it is even seen as of greater help for communication effectiveness. This is because this vision makes the teams' members very clear about what they want to communicate about (i.e. clearer and more understandable message). Case A2 formulated that as: *"EA vision is very helpful because it gives them the directions we take to deliver our project. They have many questions and if EA was not there, answering these will not be effective. So, EA vision… makes the communication clearer and more understandable"*.

For GDAD projects, it is especially important to minimise the need for communication between GDAD teams and to maximise the communication within on-site team. This also can be achieved by sharing the EA vision. Moreover, the architecture owner can help in increasing the effectiveness of GDAD communication and decreasing the frequency of communication. Case F mentioned that: *"To have an architect or architecture representative for each team is something very helpful for clear communication in specific and for development in general"*.

Some interviewees reported that they use show cases as a way of communicating problems and solutions related to GDAD projects. Show cases were seen as mechanisms for internal and external communication between all stakeholders. The show cases are seen to have a positive impact on communication process and delivering the GDAD projects. Case I reported that: *"Plans and estimations usually help to set goals and show cases help in how to achieve these estimations and plans; so, everybody is working towards fixed aim and ensure things are done in the right way"*.

8.6.7 Relationship between Agile EA and GDAD Performance

This section answers RQ4. Although the scope of EA, by definition, is to join different parts in the enterprise, given realistic time and resources, EA must focus on achievable and meaningful deliveries. This means that agile EA should evolve with the development process. Accordingly, the scope and objectives also evolve over the time. This evolution may be due to customer requirements changes, business changes, market changes, team members' behaviours, or information and technologies. Case C mentioned that: *"Scope and technology parts are highly impacted by the business"*. Moreover, agile EA artefacts should support the IT and business-planning domains, such as by providing a description of the organisation and its requirements for vendors. EA also can be used to evaluate IT and project portfolios in terms of their contribution to business goals. However, the role of enterprise architect in agile development is not to identify or force teams to use certain recourses or applications; rather the role is to provide the teams with the different available options and the estimations to deliver a certain project. This was reported by Case A2 as: *"When starting any project, enterprise architect helps providing some options for the teams depending on our estimation. Estimations may change little bit during delivery according to risks. Whenever teams need help, we give them our feedback (as architects) actively or proactively. However, if the teams can solve it themselves, we don't involve ourselves"*.

Moreover, the role of architecture owner is continuous side by side with the project iterations. This is very important in large organisations where they involve a team of architects for one project. Enterprise architect provides the estimation for resources available, time, and cost of the project. The architects share these estimations with their teams. The role of the architects is like a hub between executive levels and development teams. Case A3 described this role as: *"Architects team also provides the performance aspects estimations at portfolio level. Each program does its part. Those programs are born because of the EA. Our part in the project is consultation job whenever they need help in big projects"*. Moreover, the role of architecture owner is seen of high importance in solving the conflicts between different designs, which is a very common problem especially in GDAD environments. Case A2 described this part as: *"Conflict in design always happens. The good architect should be able to solve these conflicts on timely manner. He/ she should not dictate solutions to these conflicts since team*

members don't like that and would not follow. You never do that. You just give them choices, enough information, provide them with enough artefacts, use cases or whatever helps them, and freedom to choose among these options". In addition, architecture owner and EA vision, in general, track and ensure that team members develop what makes value to the project. Also, he/she should provide the required support to what he/she delivers. Case G explained that as: *"You need to make sure that if the team member will be able to maintain and provide supports to what he/she develop then he/she go ahead. If can't maintain and provide support, then it should not develop"*.

If all members working on a certain project are in the same open office place, then requirements and changes are distributed and shared relatively straightforwardly. However, the overheads in communications to get together to discuss the problems, to raise issues, to make decisions, and to find answers in GDAD environment can become very large. Consequently, project may be delivered late or above budget, and cause anxiety among teams and leaders. Therefore, sharing EA vision and artefacts ensure that all GDAD teams are on the same page and, even if they do not achieve high efficient and effective communication to share the goals and scope of the project, they still code without leaving the track. Consequently, they deliver on time and budget, and as per functionality and quality required (to the best of their ability). Case I reported that: *"For the last 7 years I worked in (some start-up, some big companies), on-time completion does not happen! EA would help to minimise how much you will miss the deadlines; it helps to deliver robust deliverables"*.

However, other experts do not see that good impact of using EA vision on the quality or functionality of the project delivered in GDAD environment. Case H formulated that as: *"It will achieve on-time completion and on-budget but may impact the effectiveness of functionality and quality"*. Moreover, Case I mentioned that sometimes the management level does not give the development team the opportunity to share in providing the solution architecture or EA, in general. He added, they just focus on developing the product only, which "*in my opinion it makes things worse"* (Case I).

8.6.8 Relationship between Communication Efficiency and Communication Effectiveness

As it was discussed previously in this chapter, the findings show some confusing level among interviewees regarding the communication efficiency, in general. Most of them define communication efficiency as an aspect or factor of effective communication. However, few of them clearly identified the negative tension between communication efficiency and communication effectiveness, as it was reported by Case B *"you can rush on what you say to those distributed in other countries, but you will be excluding things or team understands"*.

8.6.9 Relationship between Communication Efficiency and GDAD Performance

This section answers the first part of RQ5. Communication efficiency was interpreted to support the resource definition among GDAD teams. It is important to efficiently communicate how to share the resources' allocation (e.g., scheduling of tasks in GDAD). Case H mentioned that *"priorities were determined and quickly communicated by the product management. Also, necessities in cases regarding production issues were found"*. This reflects on the velocity and cost of project delivered.

Moreover, communication efficiency was important to solve the common issue of conflicts in design among GDAD teams. Daily meetings, for example, were assumed to increase the efficiency of GDAD communication and solve such a conflict. This conflict is reflected negatively in delivering the whole project regarding time, cost, functionality, and quality. So, communication will help to decrease or overcome the gap of project's performance if it is done efficiently. In addition, communication efficiency helps sharing project's information and requirements, which are important to ensure that every member delivers his/her tasks on-time. Case C formulated this as: *"In fact, I have to check with all stakeholders to find out if the design is good, which impacts the project velocity. The other thing that slows down the velocity is if some teams are slower than other teams or if they are not sure about the meaning of something or how it is related to the whole project"*.

8.6.10 Relationship between Communication Effectiveness and GDAD Performance

This section answers the second part of RQ5. Effective communication is another important enabler of project performance in GDAD environment. In fact, all cases mentioned that they establish a weekly (virtual) meeting among all GDAD teams to keep all teams updated with the whole project progress. The common implication for these meetings success is the effectiveness of communication between GDAD teams. Case E formulated that: "*Effective communication is very positive tip when communicating with distributed teams"*. To increase the effectiveness of the communication in large GDAD projects with all stakeholders, iteration manager represents the source of information that communicates and updates all stakeholders with the progress of the project. This will keep them on the same page and keep the project on the latest requirements. In other words, this will keep the performance of such a project on the predefined level required. The role of iteration manager was seen of significant help to both business and technical teams when many GDAD teams and stakeholders are included. Case C reported that: *"Iteration's manager of each stream contacts with all stakeholders and keep them on track which makes it more effective rather than me (integration manager) meets with that number of stakeholders at the same time"*.

Effective communication is the way to keep the tie among GDAD teams. It is the way to track the available resources for each team at any time during the project iterations. This communication enables the distributed teams to share or change the expertise, which reflects on the time, the cost, the quality and the functionality of delivered project. This is clear when a team in Sydney for example, needs an analyst at the middle of a project, which means extra time and cost will be added to the project if they have to hire a new analyst. Case E explained that:

> *The way the team was built such that resources for each team are dedicated. There is a lot of dependence. To make sure there is a continuous and constant communication and ties between the three streams we have paddles such that all three teams get together to talk and things like development paddles. This is so critical in such a large project. Although*

> *there are dedicative resources for each team, there should be a strong communication between them, and also we do share resources and help each other as much as we can.*

The need for such effective communication between GDAD teams is even more critical when the time-zone difference is big between the teams. This will add extra cost and time for travelling when experts travel from one team to another. So, if the communication was not effective enough between teams, extra time and cost will be added to the project and may result in trading-off with the project's functionality or quality. *"It is very critical when the time-zone between team members is very big. It is too hard to organise especially if it is out of core hours of teams. It is a risk for a project"*. Case H

8.7 Chapter Summary

This chapter discussed the qualitative findings of the research. The chapter represents the effort of an extensive and in-depth exploration of the textual data of the semi-structured interviews and the comments parts in the survey. Some conflicts or disagreements between the interviewees were identified and discussed in the findings. Firstly, some conflicts were identified regarding using, defining, documenting, and sharing agile EA in the context of GDAD. Secondly, some conflicts were identified regarding the role and definition of communication efficiency and communication effectiveness in GDAD. Moreover, several key issues were identified and thoroughly discussed regarding communication and using agile EA in GDAD environment and their impacts on GDAD performance.

As for survey findings, the case studies findings revealed positive significant relationships between using agile EA and both efficient and effective communication in GDAD environment. Moreover, while the survey findings showed that agile EA has significant relationships with on-budget completion, functionality and quality, and does not have any significant relationship with on-time completion, the case studies findings revealed that agile EA has significant relationships with quality and functionality, and it may enhance on-time and on-budget completions; however, trade-off between on-time

or on-budget and quality or functionality may take place. Moreover, while the survey findings showed a positive significant relationship between communication efficiency and communication effectiveness, the case studies findings revealed that the relationship is significant and could be positive or negative.

In addition, the survey findings showed that the relationships between communication efficiency and on-time completion, on-budget completion and quality were all positive and significant, and the relationship between communication efficiency and functionality was not significant. However, the case studies findings revealed that all relationships between communication efficiency and on-time completion, on-budget completion, functionality and quality were positive and significant. Furthermore, the survey findings showed positive significant relationships between communication effectiveness and on-budget completion, functionality and quality, and the relationship between communication effectiveness and on-time completion was not significant. However, the case studies findings revealed that all relationships between communication effectiveness and on-time completion, on-budget completion, functionality and quality were positive and significant. The following chapter discusses the findings of this study. It synthesises both quantitative and qualitative findings and draws the cross-analysis findings. It also revalidates the research model according to the cross-analysis findings.

Chapter 9 Discussion of Research Findings

In Ch.7, the findings of the survey were discussed. This included the statistical analysis of the measurement model and the hypotheses testing using the PLS-SEM method. The statistical results showed strong support to the positive relationships between agile EA and GDAD active communication, agile EA and GDAD performance, and GDAD active communication and GDAD performance. In Ch. 8, the case studies findings showed strong support to most of the statistical analysis findings and more insights were revealed into the non-supported hypotheses. However, surprisingly, while some hypotheses were strongly statistically supported, they were found to be not strongly supported from the interviewees. Other hypotheses were found to be supported more or less in the interviewees than in the statistical results. Therefore, this chapter synthesises both findings and draws additional insights about the research model (theory), and links these findings and insights back to the original research questions.

9.1 Introduction

This chapter answers research questions (RQ3-RQ6) by discussing the key insights of the thesis through synthesising the findings from both survey (Ch.6 and Ch.7) and case studies (Ch.8), in the lenses of the available literature. This triangulation makes the study findings more valid and rigorous.

The discussion in this chapter follows the same structure (as in Ch.7 and Ch.8) by discussing the model's constructs and relationships through which we answer the research questions. This chapter is organised as follows. First, in Section 9.2, the descriptive findings of the survey respondents are discussed and a summary of PLS-SEM findings are outlined. Section 9.3 synthesises the findings of both survey and case studies, and the previous studies' findings. Section 9.4 presents some recommendations based on the cross-finding results. Then, reflectional learning is presented in Section

9.5, which discusses the validation of the research model (theory). Finally, the chapter summary is presented in Section 9.6.

9.2 Summary of the Survey Findings

This section discusses the main findings obtained from analysing the demographics of the survey respondents, and summarises the PLS-SEM findings. Five demographic variables, namely; GDAD experience, number of teams, number of team members, outsourcing approach, and number of vendors (providers) were subjected to frequency analyses. The aim of this analysis was to find out any significant findings or relations between these demographics and using agile EA as communication and performance enabler, or between these demographics and active communication as a high-performance enabler in GDAD environment.

9.2.1 Descriptive Findings

Most of the respondents had more than (5-10) years' experience in GDAD (40%) or had experience of (2-4) years (38.8%). Most of respondents (27.5%) were team leaders/Scrum Masters or agile coaches (15.6%). Most respondents (40.6%) worked with (3-5) teams or (25%) worked with (6-10) teams. Most of the teams (25.6%) included (4-10) members or (23.8%) included (11-20) members. It can be concluded that respondents who have long experience in GDAD, who are in leading positions, and who usually work with many teams and with many members among the teams are more likely to realise the impact of agile EA on GDAD performance. They are also more likely to realise the importance of agile EA and active communication on the overall project performance in GDAD environment (Martini et al. 2013).

Moreover, most respondents' organisations got few, some or significant outsourced projects (93.1%). It is very clear that those organisations that employ an outsourcing approach are very supportive of the use of agile EA with their providers as a communication and performance enabler. This may be because these organisations try to avoid the common outsourcing issues by using agile EA. These issues involve lack of clear specifications and internal support, over optimistic expectations, and selecting the wrong provider (Bush et al. 2008; Carmel & Abbott 2007).

In addition, most respondents' organisations approach in choosing their vendor was using only one vendor (43.1%) or the best available vendors (40.6%). It can be concluded that either employing only one vendor or different vendors, GDAD organisations prefer using agile EA and see it as an enhancer to their communication and overall performance project. Agile EA may help vendors (especially new vendors) to succeed since it increases the balance between optimising the vendors process with the undertaking changes when working for another organisation, which may prevent unnecessary investments and painful performance failures (Moe et al. 2014).

9.2.2 PLS Model Findings

The statistical analysis (Ch.6 and Ch.7) indicated that the measurement model had demonstrated satisfactory reliability and validity (i.e. according to EFA, Cronbach's alpha, composite reliability, item cross-loading, and AVE tests). Therefore, the 26 items were retained and were able to measure the research model constructs. The structural model was then tested using SEM technique which was recommended to test IS explanatory attitudinal research (Gefen et al. 2000), and more precisely, PLS-SEM was chosen. PLS has the ability to handle both reflective and formative constructs, it demands fewer sample size comparing to other models, and it is more appropriate for empirical research than CB-SEM models for exploratory research (Hair et al. 2014; Lowry & Gaskin 2014).

Table 9.1 summarises the hypotheses testing results. Ten hypotheses were supported form the statistical analysis. Five hypotheses were not supported. It is important to mention that two hypotheses (i.e. communication efficiency→communication effectiveness and communication effectiveness→on-budget completion) were rejected due to the sign of β (opposite to what it was hypothesised) values. In other words, the relationships were significant, but they were positive.

Figure 9.1 shows R^2 values, the significant relationships in the model, their directions (the direction of the arrow identifies the relationship direction), and the strength of each relationship. All relationships were found positive. Further findings are worth

mentioning, as shown in Figure 9.1 and Table 9.1. First, the results show that agile EA has a very large effect on communication efficiency which is stronger than the effect on communication effectiveness. Second, communication efficiency has a very large effect on on-time completion, while communication effectiveness has very large effects on both functionality and quality.

Table 9.1: Summary of the research hypotheses testing results

Hypothesis	Relationship	Hypothesis Supported
H1a	Agile EA → Communication efficiency	Yes
H1b	Agile EA → Communication effectiveness	Yes
H1c	Agile EA→ On-Time completion	No
H1d	Agile EA→ On-Budget completion	Yes
H1e	Agile EA→ Quality	Yes
H1f	Agile EA→ Functionality	Yes
H2	Communication efficiency → Communication effectiveness	No (positive β)
H3a	Communication efficiency → On-Time completion	Yes
H3b	Communication efficiency → On-Budget completion	Yes
H3c	Communication efficiency → Functionality	No
H3d	Communication efficiency → Quality	Yes
H4a	Communication effectiveness → On-Time completion	No
H4b	Communication effectiveness → On-Budget completion	No (positive β)
H4c	Communication effectiveness → Functionality	Yes
H4d	Communication effectiveness → Quality	Yes

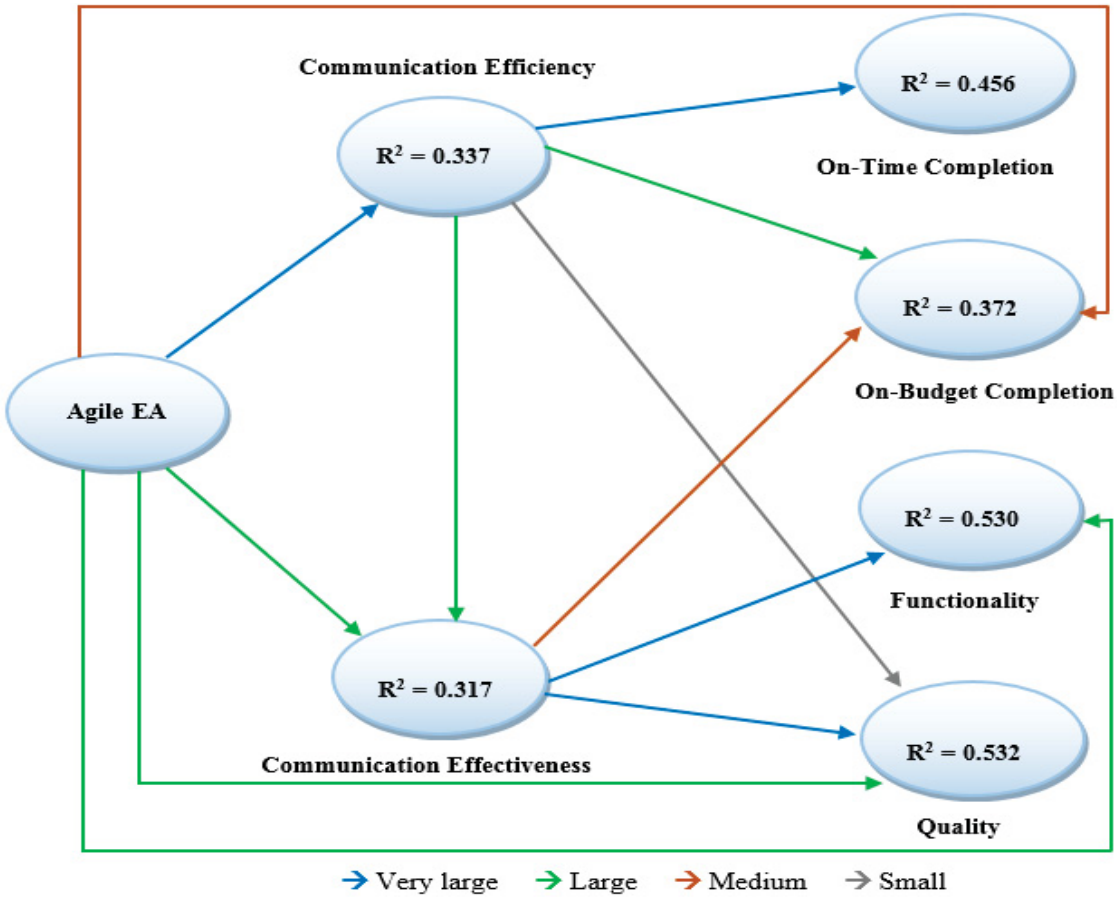


Figure 9.1. Strength of significant relationships in the research model

Table 9.2 summarises the mediation role of all relationships in the structural model. The relationship between agile EA and on-time completion was fully mediated by communication efficiency. In addition, communication efficiency partially mediated the relationship between agile EA and on-budget completion. Moreover, communication effectiveness partially mediated the relationship between agile EA and quality. However, the relationship between agile EA and functionality was not mediated by either communication efficiency or communication effectiveness. Moreover, communication effectiveness was found to fully mediate the relationship between communication efficiency and functionality and partially mediate the relationship between communication efficiency and quality.

PLS-SEM does not have a standard goodness-of-fit statistic (Hair et al. 2014). Instead, the assessment of the model's quality is based on its ability to predict the endogenous constructs using R^2, f^2, and Q^2 values. According to the values of R^2, f^2, and Q^2, the research model can be characterised as of moderate level of predictive accuracy.

Table 9.2: Summary of mediation effects

	AEA*EFFIC			AEA*EFFECT			EFFIC*EFFECT		
	Effect (β)	Sig.	Mediation Type	Effect	Sig.	Mediation Type	Effect	Sig.	Mediation Type
EFFECT	0.041	ns	No mediation						
TIME	0.151	**	Full mediation	-0.101	ns	No mediation	0.035	ns	No mediation
BUDGT	0.255	***	Partial mediation	-0.108	ns	No mediation	0.079	ns	No mediation
FUNC	0.021	ns	No mediation	0.055	ns	No mediation	0.129	***	Full mediation
QLTY	0.107	ns	No mediation	0.143	*	Partial mediation	0.164	**	Partial mediation

*p < 0.10, **p <0 .05, ***p <0 .01.

9.3 Cross-Analysis Findings-Addressing Research Questions (RQ3-RQ5)

There is a common agreement among the agile community that communication is core to agile development and it is also the main challenge of GDAD (e.g., Ambler 2015; Boehm & Turner 2003; Gill 2015b; Leffingwell 2007; Sutherland et al. 2007). This was also supported from the case studies, as it was mentioned by Case B that "*the biggest problem we face with distributed teams is the communication"*. Moreover, our findings from the case studies clearly revealed that face-to-face is the most efficient and effective communication channel. This is also strongly supported in the literature (e.g., Agile Manifesto 2001; Cockburn 2007; Highsmith 2009; O'Leary et al. 2014; Pikkarainen et al. 2008; Schwaber 2004).

Despite the long-lasting debate between business representatives and technical representatives (e.g., development teams) in agile development about the value of using EA in such a very flexible environment, there is no doubt that agile EA role is essential for any agile development organisation these days (Op't Land et al. 2008; Proper &

Greefhorst 2011). Not like traditional SD, "agile projects tend to want to evolve the architecture, as needed, in real time, whereas large software projects have traditionally favoured considerable up-front analysis and planning" (Bass et al. 2013, p. 279). This role is even more important in GDAD environment (Kornstädt & Sauer 2007; Sauer 2006). It has been suggested that the holistic and integrated views of agile EA provide the "possibility to see and discuss how different parts (the ICT systems, the processes, etc.) are interconnected and interplay. Understanding means not only knowing what elements the enterprise consists of and how they are related from different aspects, but also how the elements work together in the enterprise as a whole" (Karlsen 2008, p. 219). As a part of this role is its impact on communication process and the project's performance level in GDAD (Kornstädt & Sauer 2007), which is the focus of this study.

Statistical analysis showed that agile EA, communication efficiency, and communication effectiveness collectively explained 45.6 percent of the variance in "on-time" completion, 37.2 percent in "on-budget" completion (medium predictive power), 53 percent in software functionality (moderate high predictive power), and 53.2 percent in software quality (moderate high predictive power). Moreover, when we modelled the performance as second-order model, agile EA, communication efficiency, and communication effectiveness collectively explained 66 percent of the variance software performance (substantial predictive power). These findings of high predictive power indicate that these three constructs represent a high value to GDAD and they highly impact the performance.

In the following sections, we answer the research questions (RQ3-RQ5) by synthesising and discussing findings from both survey and case studies in order to draw a clearer picture of that role. Table 9.3 summarises the cross-analysis findings. This discussion will be also supported by the previous studies' findings from the literature to give more generalisability to our findings.

Table 9.3: Cross-analysis findings of the survey and case studies

Research Question	Hypothesis	Survey Support	Case Study support
RQ3: How does agile EA affect GDAD active communication?	Agile EA positively affects the efficiency of the GDAD communication	Large support	Supported
	Agile EA positively affects effectiveness of the GDAD communication	Large support	Large support
RQ4: How does agile EA affect GDAD software performance?	Agile EA positively influences GDAD software performance (on-time completion, on-budget completion, software functionality, and software quality)	The influence on on-time completion was not supported. All others were supported	All hypotheses were supported. However, trade-off between on-time and on-budget, and quality or functionality may take place
	GDAD communication efficiency negatively affects effectiveness of the GDAD communication	Not supported	Partially supported
RQ5: How does GDAD active communication affect GDAD software performance?	Communication efficiency positively influences GDAD software performance	The influence on functionality was not supported. All others were supported	All hypotheses were supported
	Communication effectiveness positively influences functionality and quality, and negatively influences on-time completion and on-budget completion	The influence on on-time was not supported. The influence on on-budget completion was not supported (significant but positive). The influence on functionality and quality was supported	The influence on functionality and quality were largely supported. The influence on on-time completion and on-budget completion was not supported (the influence was found large but positive)

9.3.1 Effect of Agile EA on GDAD Active Communication (RQ3)

This section answers RQ3. EA can be used as a communication tool with a more generic purpose, for instance in business–IT or organisational alignment (Tamm et al. 2011), or for sharing an overview of the organisation and its objectives among GDAD teams (Helquist et al. 2011; Niemi & Pekkola 2015). Previous studies emphasise the role of indirect communication in the GDAD such that the architectural and requirements specifications should be documented at a high level of abstraction (Selic 2009; Stettina et al. 2012). Direct communication may, therefore, not always be the best

choice and may also become an inhibitor (Vidgen & Wang 2009). Using agile EA view (i.e. employing central digital repository), which is accessed by all team members, with the available communication tools/media allows all team members to view and manipulate the story boards in real-time (Maruping 2010; Pikkarainen et al. 2008). Agile EA view is not a specification for implementation per se; instead, EA artefacts provide high-level guidance for individual development projects. This view helps in receiving early feedback to deal with any change or intervention needed and in maintaining early decisions and clear focus of agile team (Buckl et al. 2011). This view serves as a communication platform with a common vocabulary that is stated to improve mutual understanding and communication effectiveness (Sharp & Robinson 2007). The lack of appropriate architecture represents a communication barrier through misunderstanding or an unnecessary flow of communication due to the insufficient definition of a system and software structure (Martini et al. 2013). The socio-technical congruence measure was introduced by Cataldo et al. (2008) to measure the alignment of task dependencies to developer communications. When the alignment was high, the communication was found to be more efficient and more effective (Cataldo & Ehrlich 2011).

As shown in Figure 9.1 and Table 9.1, the statistical analysis showed a very strong relationship between agile EA and communication efficiency. Agile EA explained 33.7 percent of the variance in communication efficiency (above medium predictive power). This means that using agile EA vision among GDAD teams can be used as a mechanism or strategy to increase the efficiency of the communication. This finding was also supported by the case studies as it was clearly reported by Case E2: *"My Company has a repository that stores all solutions, which includes all these diagrams and charts, tables, and use EA artefacts...this helps a lot to speed up the conversation with distributed teams, especially off-shore teams"*. Agile EA repository was recommended as a mechanism that increases the efficiency of GDAD communication (Gill & Bunker 2013; Maruping 2010; Valimaki & Kaariainen 2008). In addition, this repository involves all solutions, which can be used in refactoring to produce simple design. This design influences communication indirectly by enabling simplicity, which is needed to communicate efficiently (Fruhling & De Vreede 2006; Bosch & Bosch-Sijtsema 2010; Pikkarainen et al. 2008).

Moreover, the statistical analysis showed a strong relationship between agile EA and the effectiveness of GDAD communication. Agile EA and communication efficiency collectively explained 31.7 percent of the variance in communication effectiveness (above medium predictive power), also. This also indicates that using agile EA vision can enhance the effectiveness of communication among GDAD teams. This finding was also supported by the case studies. For example, Case A2 reported that *"EA vision is very helpful...it makes the communication clearer and more understandable"*. This finding was also supported from the previous literature. The lack of appropriate architecture decreases the knowledge sharing and communication effectiveness among global agile development teams and team's members (Jaanu et al. 2012). Project planning (some coordination structure) using agile EA vision, even if it is not very detailed in nature, has positive influence on communication effectiveness (Batra 2009; Yadav et al. 2007). Agile EA vision can provide terms and concepts that serve as a common language for all DAD teams, which enables clear communication and arrangements (Kornstädt & Sauer 2007). Moreover, using this vision by all GDAD teams and members, especially at the beginning of the project, can increase the effectiveness of communication (Martini et al. 2013).

9.3.2 Effect of Agile EA on GDAD Performance (RQ4)

This section answers RQ4. As shown in Table 9.3, we experienced some differences between the survey findings and the case studies findings. The survey findings showed statistical support to the positive relationships between agile EA and GDAD performance aspects (i.e. on-budget completion, functionality, and quality), and non-significant with on-time completion. However, the findings of case studies showed confusing role of agile EA on GDAD performance. While business side assume that EA helps in meeting all performance aspects, technical (e.g., development) teams see the focus of EA is more on quality and functionality. In general, the impact of using agile EA was more recognised and supported among the interviewees on functionality and quality of the delivered project in GDAD. This may be due to the fact that the off-shore teams (i.e. providers or contractors) have to agree on the cost and time of delivery at the beginning of a project, as it was reported by case C: "*They have to deliver on time and cost, as per agreement in the beginning of the project*".

The previous literature has also shown support to the impact of EA on the project performance. Agile EA provides the basis for architecture rules, which improves implementation consistency, reduces the number of errors, leverages same design patterns and language patterns, and scores from same quality attributes and uses a uniform scoring system (Kornstädt & Sauer 2007; Madison 2010). The conflict between teams about solution design can be addressed by using architectural description (Martini et al. 2013). Agile EA vision provides a plan for task allocation, facilitates the cooperation of GDAD team members, and enables continuous integration reaching across distributed teams (Sauer 2010). Agile EA vision helps especially in defining project functionality and interrelationships with the environment (Helquist et al. 2011).

Using agile EA vision guidance combined with training may be facilitating knowledge improvement and productivity of developers (Van Waardenburg & Van Vliet 2013). This awareness may enable GDAD organisations to produce software with the cost, schedule, and quality goals (Green et al. 2010). Moreover, quality has positive correlation to timeliness such that late projects are unlikely to achieve a good quality; however, trade-off on quality is often accepted to meet deadlines (Estler et al. 2014). Bass et al. (2013) went further than that and claimed that "Fortunately, it is possible to make quality predictions about a system based solely on an evaluation of its architecture" (p. 28). Other studies reported that the socio-technical congruence (measure of the alignment of task dependencies to developer communications) has a positive impact on project performance (Cataldo et al. 2008). The high level of socio-technical congruence results in higher quality and quicker delivery (Ramasubbu et al. 2011).

9.3.3 Relationship between GDAD Active Communication Dimensions (Efficiency, Effectiveness)

The statistical analysis showed strong positive (not negative as it was conceptualised) relationship between communication efficiency and communication effectiveness, as shown in Figure 9.1. This positive impact may be due to the way the questions in the survey were asked. In addition, the respondents may refer to communication efficiency

as to efficiently establish communication only and not communicating efficiently also, which in this way leads to more effective communication. This is true since efficiently establishing communication and information requirement provides effective communication (Daft et al. 1987; Dennis et al. 2008).

Moreover, some case studies findings showed that the negative relationship between communication efficiency and communication effectiveness is supported. For example, Case D reported that "*if you rush and do not take your time, or you do not ask the question in the right way, you will have misunderstanding to your message*". However, other cases showed a high level of confusing or mixing up between these two dimensions. The relationship between the two dimensions was not clearly recognised. It was clear that most interviewees refer to both communication efficiency and communication effectiveness as communication effectiveness (i.e. effectiveness of the communication does make the difference). In other words, they define communication effectiveness in terms of clearness and quickness of the message, which is conflicting with our conceptualisation of communication efficiency and communication effectiveness. Some literature supports this view and explains that communication effectiveness facilitates knowledge transfer rapidly between GDAD teams (Dorairaj et al. 2011).

While our cross-analysis showed a high level of ambiguity about the relationship between communication efficiency and communication effectiveness, previous literature supports our conceptualisation of this relationship. Due to GDAD communication challenges (e.g., language and culture), the message may not be received as it was intended. The shortness may be insufficient to deliver clear message (Dennis et al. 2008; Hinds & Mortensen 2005). Moreover, the higher communication effectiveness comes at the price of considerably longer time (Dyba et al. 2007). Communication efficiency may decrease the development of understanding since GDAD teams or team members will not have the time required to fully process the information, which may cause a greater cognitive load on the member and encourage untimely action (Dennis et al. 2008; Te'eni 2001).

9.3.4 Effect of GDAD Active Communication on GDAD Performance (RQ5)

This section answers RQ5. As shown in Figure 9.1, the statistical findings showed very large impact of active communication on project performance in GDAD environment. While communication efficiency impact was stronger on on-time and on-budget completion, communication effectiveness impact was stronger on the functionality and the quality of GDAD project. Although not all relationships were significant, our conceptualisation hypotheses were partially supported in that communication efficiency and communication effectiveness differentially impact GDAD performance aspects.

Also, it was highly recognised from all cases that communication is behind the success of GDAD. Case E reported that "c*ommunication is very positive tip to project success within distributed teams"*. Although the role of communication effectiveness was recognised more than the role of communication efficiency, both of these two dimensions share the importance for GDAD performance. Both dimensions were important to share the resources among GDAD teams, to keep all teams on the same page, and to cut down the cost and time of involving new resources, which are reflected on the time, the cost, the quality and the functionality of GDAD project.

The previous literature strongly supports that communication is behind the success of agile development (e.g., Boehm & Turner 2003; Gill 2015b; Leffingwell 2007; Sutherland et al. 2007). Highsmith and Cockburn (2001) explain that communication within agile teams is vital to meet cost, schedule, quality, and functionality goals. Communication efficiency and communication effectiveness were found as key determinants of project performance in large complex agile development organisation (Helquist et al. 2011; Piccoli, Powell & Ives 2004; Pikkarainen et al. 2008).

Agility in GDAD is the capability of a GDAD team to speedily accomplish tasks and to adapt and reconfigure itself to changing conditions in a rapid manner (Sarker & Sarker 2009). Also, Korkala et al. (2010) stated that agile practices are the most successful in the environments where rapid communication is enabled. The inefficient GDAD communication leads to delay of information flows, delays in completing tasks,

misalignment and rework (Korkala & Abrahamsson 2007; Misra et al. 2009; Sindhgatta et al. 2011).

Effective communication is important for software development teams to facilitate knowledge transfer between team members, to allow team members to understand the requirements from clients and to help team members perform development activities efficiently (Dorairaj et al. 2011; Green et al. 2010). The use of an adaptive method together with effective communication tools, under optimum conditions, can create a high quality GDAD and resulting systems (Gill & Bunker 2013; Green et al. 2010). In order to achieve the benefits in GDAD, communication between the distributed parties must be effective (Holmström et al. 2006b; Korkala & Maurer 2014).

9.4 Recommendations for Using Agile EA and Enhancing Active Communication

After synthesising the findings of both quantitative and qualitative analysis, and previous literature, we came up with common recommendation regarding using agile EA and communication to enhance a project's performance in GDAD environment. High levels of ambiguity in GDAD environment can be addressed through increasing the Common Ground (awareness and knowledge) by using agile EA vision, which will increase the efficiency and effectiveness of communication among the communicative teams, as it was shown in this study. These high levels of shared knowledge and communication between the GDAD teams result in higher software project performance level (Estler et al. 2014; Helquist et al. 2011). We believe that established practices and tools of GDAD can be employed with some sort of formalisation and structure (using agile EA vision) without losing the flexibility of agile practices (Kornstadt & Sauer 2007; Sauer 2010).

9.4.1 Agile AE

The tension between employing agile EA in agile development is still arguable between business and development teams. The definition and role of agile EA or how to employ EA within agile development approaches still warrants more work, as it was mentioned by most of the interviewees and respondents, for example R8 that "*EA is applied at*

strategic level and agile at execution level". This is not what agile EA is or what it should be from agile development point of view. Using agile EA imposed on the process needs to be flexible enough that team members or team leader can choose a particular course of action based on the outcomes of the ongoing collaborative development (Helquist et al. 2011). As long as agile EA is still defined and practiced in the traditional way, the benefits of using it will be negative on the development teams and on the project performance. To sum up, agile EA needs to be part of the evolution and emergent design and planning during the project life cycle, and it should be communicated in both directions (executive and development team) regularly to achieve the aim. Agile EA development "should be seen as an iterative process, not as an up-front task that establishes unalterable structures" (Sauer 2010, p. 327). We recommend the following when using agile EA in GDAD:

- Enterprise architect (s) and architects' teams usually underinvest the big effort of communication EA. The first step in employing agile EA is to communicate it throughout the teams, explain its benefits, and inform team members on how to use it.

- A common vision of agile EA with simple and clear artefacts (vocabulary, tables, diagrams, high level design, and so on) should be created and marketed to the development teams. This vision is saved and updated in a central repository that can also have cases, questions and answers, tracking data, calendar of events, and team and personal blogs.

- Agile EA vision provides uniform standards, checklists and examples. It also provides the common framework and the multiple options to deliver the software.

- The core architecture of the application is built and solidified by the main on-site team (most experienced members) during the first iteration. This way, off-shore teams (e.g., developers) can start up building on a solid architectural base.

- Enterprise architect/ architecture owners should be involved with other teams during development iterations. They should provide help and resolve misunderstandings between the teams.

9.4.2 Communication Efficiency and Communication Effectiveness (RQ6)

This section answers RQ6. The tension between communication efficiency and communication effectiveness is yet to be recognised by GDAD practitioners. While information should be always effectively shared, sometimes a message should be quickly shared which may impact the effectiveness level. In general, GDAD communication needs to be quickly (i.e. both ways communication channels should be instantly available) established and effectively shared. We recommend the following when establishing communication between distributed teams in GDAD:

- High Common Ground among GDAD teams increase the shared knowledge and understanding about the project. This can be done by using agile EA vision that is saved in a repository and shared or communicated with all GDAD teams. This decreases the need for frequent communication and increases the efficiency and effectiveness of communication in GDAD environment.

- Different communication tools or applications between GDAD teams should be employed and used. These tools could be of high help: video-chat, Wiki and instant messaging applications, virtual whiteboards, and emails.

- Dual-shore development structure can be very helpful for higher level of active communication with on-site main team (works as a hub) that splits the work between off-shore teams.

- All teams' members (especially new team members) should be introduced to other teams (e.g., through a web conference). Training and teaching different cultural teams about other cultures is very helpful to GDAD active communication.

Moreover, team leaders should ensure that all members are trusted and equal, should encourage the common team spirit among GDAD teams.

- Exchanging team members between different locations in order to know their co-workers, their working conditions, and to develop a better understanding of their tasks and culture will enhance active communication.

- Team members in different time zones should adjust their working hours to reach maximum overlap. Team leaders (on-site main team) should find the more suitable time and date for overlapping, taking into account other cultural teams' holidays or lunch time, for example.

- Stand-up meetings (e.g., Scrum meetings) which are held every day with a video conference system and other meetings which are scheduled weekly and monthly enhance level of active communication.

- "Proxy customers" who take the role of real customers are helpful for high level active communication. They should have sufficient knowledge in the application domain and should be able to interact equally well with technical and business project teams.

9.5 Reflectional Learning (Main Research Question)

After discussing the synthesised results from survey findings and case studies findings, and the supporting literature to our findings, we re-visit our theoretical conceptualisation of this study. This was also done by extracting the commonality between the GDAD communication challenges theoretical framework (Figure 3.2), the research model (Figure 4.5), and the findings of this chapter and recommendation (Section 9.4). The resulting model, as shown in Figure 9.2, represents the validated version of theory of this study. It is important to indicate that recommended strategies, techniques, and agile EA vision are used side by side with available communication tools such as TV-links, face-to-face, emails, phone, and so on.

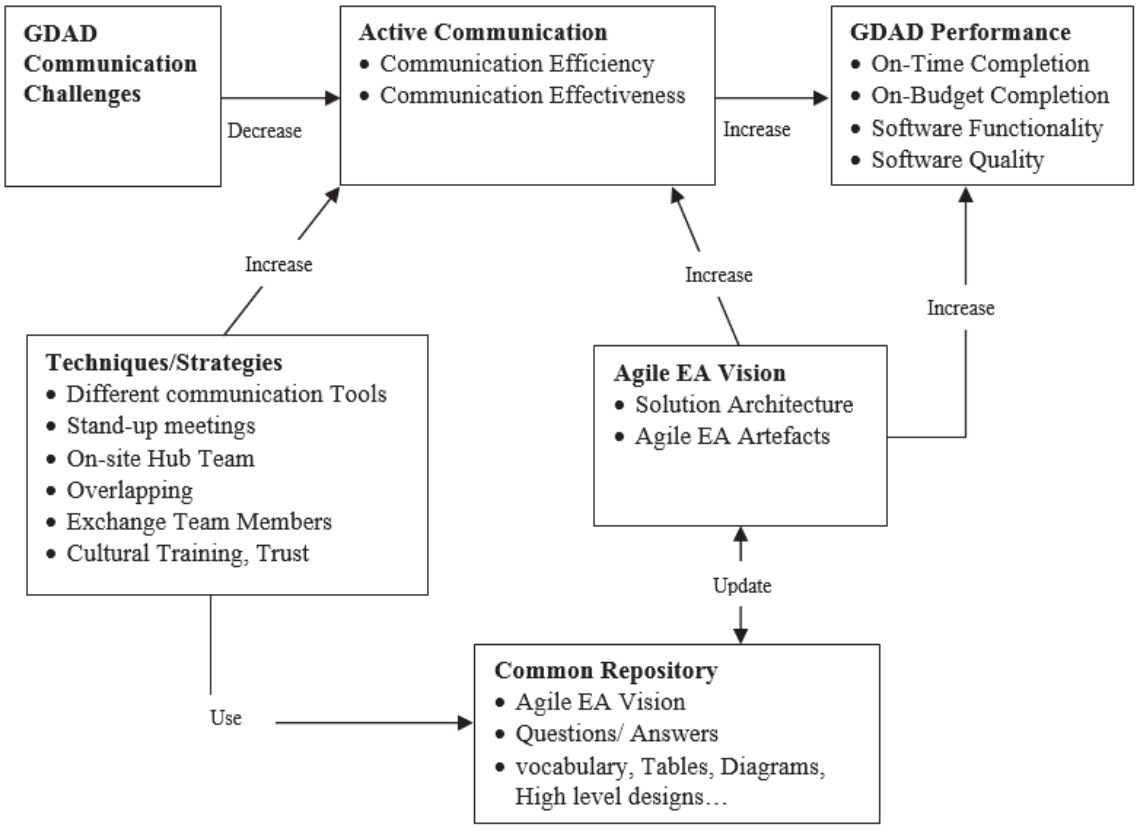


Figure 9.2. Theory fine-tuning

9.6 Chapter Summary

This chapter synthesised the findings from both survey and case studies, in the lenses of the available literature. While, in general, all relationships between agile EA, active communication dimensions, and software performance aspects were supported in the synthesised findings, some lack of agreement and understanding is still there when using agile EA. This lack of agreement and understanding is due to defining and communicating agile EA. It is important for EA in agile development to be agile (flexible and evolve with the iteration). Moreover, it is important to share the value of agile EA with all teams and members in order to achieve its benefits. Agile EA was found to positively impact active communication and software project performance in GDAD environment. In addition, active communication was found to be strongly related to performance in GDAD environment. It is important to quickly establish GDAD communication and it should be effective.

The next chapter concludes this study by providing the main findings of this research. This chapter also discusses the contributions of this research to theory, practice, and teaching and learning. Furthermore, it identifies a number of limitations and provides some directions for future research in the GDAD communication context.

Chapter 10 Conclusion

This study has investigated the impact of agile EA on active communication and on software performance, and the impact of active communication on software performance in GDAD environment. We have built a research model based on the extensive literature review (Ch.2, Ch.3 and Ch.4). This research model has been tested by conducting a survey (Ch.6 and Ch.7) and semi-structured interviews with 10 post hoc case studies (Ch.8). Both findings of the survey and case studies (Ch.9) revealed that agile EA had positive impacts on both communication efficiency and communication effectiveness, and a trade-off between on-budget completion and on-budget completion with software functionality and quality. Moreover, communication efficiency and communication effectiveness had a strong positive impact on all software performance aspects. This chapter provides a summary of the key findings and contributions of this study, and suggests some directions for future research.

10.1 Introduction

The purpose of this chapter is to conclude the thesis by highlighting the main contributions and limitations of this research, and future research directions. Section 10.2 presents an overview about the original research problem and motivations of this thesis. Section 10.3 discusses the findings of this study (from both survey and case studies findings). Section 10.4 outlines the main contributions of this study to research, theory, practice, and teaching and learning. The limitations of the study are outlined in Section 10.5. The avenues for further research opportunities arising from this study are presented in Section 10.6 before concluding in Section 10.7.

10.2 Overview of the Study

Agile Manifesto states that the "*most efficient and effective method of conveying information to and within a development team is face-to-face conversation*" (Agile

Manifesto 2001). Existing research supports the notion that having the entire development team co-located (i.e. work in the same physical location) vastly improves communication efficiency and communication effectiveness (Cockburn 2007; Highsmith 2009; O'Leary et al. 2014; Schwaber 2004). Moreover, informal and face-to-face communication is important for an agile development team during the entire development lifecycle.

Nevertheless, co-located ASD is not always the case. In early 2000, it was indicated that around one million of the jobs are in IT-related fields (Engardio et al. 2003), and Forrester estimated that approximately 3.3 million jobs and $136 billion in salaries are expected to shift overseas by 2015. This was, perhaps, due to the increasing trends of outsourcing and geographically distributed development for the perceived benefits of short time to market, reduced production cost etc. Thus, similarly, the combination of geographically distributed development and agile practices seems to offer many benefits, such as low production cost, the opportunity to involve the most talented developers around the world and faster time to market. This has created an environment where ASD teams are not exclusively co-located, but operating in different regions, dispersed throughout the world (i.e. GDAD). This tendency seems to continue and the use of GDAD teams will be likely to grow, outpacing our understanding of their dynamics including the influence of communication in GDAD environment (Niazi et al. 2013).

Active communication is becoming important and challenging in GDAD because there are fewer opportunities for informal face-to-face communication and larger number of interdependent teams and projects (Gill 2015b). Although, it has been reported in literature that active communication may enhance GDAD design and quality by reducing the IS project development time and cost, the empirical knowledge on the subject seems to be scarce. There is a need to empirically examine how active communication can be achieved to enhance the GDAD performance. To address this important gap, this study empirically investigated the impact of agile EA as a means that could enhance active communication and software performance, and the impact of active communication on software performance in GDAD environment.

10.3 Summary of the Study Findings

The aim of this study was to enhance GDAD communication. In order to build a clear picture of the GDAD active communication issue, this study first systematically reviewed the challenges of GDAD communication and identified a number of challenges and the solutions/ techniques or strategies to mitigate the impact of these challenges. Among these strategies, using EA in agile development was seen to have potential for enhancing GDAD active communication. To explain the impact of agile EA on GDAD active communication, this study put forward the following question:

How does agile EA affect GDAD active communication?

Moreover, since EA is related to deliver high performance in traditional software development, it was important to study the extent to which agile EA may impact the performance in GDAD environment. Therefore, the second question was formulated:

How does agile EA affect GDAD software performance?

Finally, to understand the extent to which active communication is behind delivering high performance software in GDAD we formulated the third question:

How does GDAD active communication affect GDAD software performance?

To answer these questions, we built a conceptual model based on the available literature; mainly we used Common Ground theory to conceptualise active communication dimensions and role of agile EA as an enabler to common ground. This model comprises three main constructs: agile EA, active communication (communication efficiency and communication effectiveness), and software performance[8] (on-time completion, on-budget completion, software functionality, and software quality). The constructs and measurements of the model were validated

[8] Similar to other studies, we collected practitioners' human perspectives, based on their experience, to measures on-budget, on-time, quality, and functionality.

through expert (both academia and industry of agile development) evaluation and pilot study. The model was then evaluated through two phases: quantitatively (using measurement-based survey technique) and qualitatively (using semi-structured interviews technique). In both phases, the unit of analysis was team member of agile EA driven GDAD organisations. In the quantitative phase, the measurement model was first validated to make sure it was appropriate for SEM validation. Then, SEM was applied to the research model. In the qualitative phase, data was collected and analysed from 10 post-hoc case studies and the comments parts from the survey.

The findings showed that all the above three questions were supported, with some little differences between the quantitative and qualitative findings. Firstly, using agile EA was statistically found to impact, to a very large extent, communication efficiency (explained 33.7% of the variance of communication efficiency) and to a large extent communication effectiveness (explained 31.7% of the variance of communication effectiveness). The qualitative findings also showed that agile EA can enhance both communication efficiency and communication effectiveness.

Secondly, using agile EA was statistically found to enhance all software performance except on-time completion (i.e. on-budget completion, software functionality, and software quality). The qualitative findings also showed that all performance aspects will be enhanced using agile EA; however, on-time and on-budget completions may be come at the cost of of quality or functionality, sometimes.

Finally, high levels of communication efficiency and communication effectiveness were statistically strongly related to delivering high software performance. However, these relationships are differential with the software performance aspects. While communication efficiency was very strongly related to on-time completion and on-budget completion, communication effectiveness was strongly related to delivering high functionality and quality of the software. The qualitative findings showed strong support to both communication efficiency and communication effectiveness on enhancing software performance.

These findings suggest that agile EA can enhance both communication and software performance in GDAD environment. Using agile EA represents more affordable and cheaper choice in GDAD without the need to include more visits' exchange or extra training. All it needs is just to communicate agile EA among GDAD teams. Moreover, the findings support the previous findings and recommendations that communication is behind delivering high software performance in GDAD environment.

10.4 Contributions of the Study

Given the importance of GDAD in today's business world and the critical issue of GDAD communication, the need for a strategic solution to GDAD communication issue is getting bigger. Our solution was by employing agile EA that would enhance both communication and software performance in GDAD environment. Hence, by developing and validating a theoretical model and the accompanying measurement instrument for assessing the effect of agile EA on communication process and software performance and the impact of communication process on software performance in GDAD environment, this study contributes to research, theory, teaching and learning, and community in several ways. This section highlights these contributions.

10.4.1 Contribution to Academic Research and Theory

As far as the contributions of this research towards academic research and theory are concerned, academic research interested in the concept of GDAD and agile EA will find merit in the work and underlying argument of this thesis in several facets. Firstly, the thesis offers a comprehensive and deepened perspective on the existing discourses on GDAD communication challenges. The researcher followed a well-known systematic literature review method to systematically identify and analyse these challenges as well as the solutions that have been proposed to mitigate these challenges. Moreover, the thesis offers a comprehensive and deepened perspective on the existing discourses on agile EA and active communication. The researcher followed a rigorous approach to analyse the literature. This study represents an original contribution to IS research and can serve as knowledge building blocks for future research on GDAD communication and agile EA.

Secondly, the thesis introduces two new agile development communication dimensions (constructs), namely; communication efficiency and communication effectiveness which have not been discussed before as explained and empirically analysed here in this study context. Hence, these two constructs could be observed separately. These constructs open up the 'black box' of communication process in GDAD research. The thesis has provided both definitions of the two constructs and measurement indicators for them. In addition, the study findings showed that the two dimensions affect differently the software project performance in GDAD environment, which advances theory.

Thirdly, the research has developed adequately validated measures that can be used in future research. For example, the measures of agile EA are original and have not been suggested in the previous literature. The measures for communication efficiency and communication effectiveness have been refined from the IS literature to capture the unique attributes of both constructs. The rigorous procedure followed to establish the psychometric properties of these measures make them suitable for use in future research.

Fourthly, the research empirically demonstrates that the strategic value of agile EA, as a whole, is an important variable that GDAD organisations (agile EA driven IS development) may apply to enhance their communication and increase their IS development project performance. Hence, this research contributes to the body of knowledge on the relationship between agile EA and active communication in particular and agile EA and software performance in general. It also supports the existing GDAD researches and recommendations by empirically providing evidence that high efficiency and effectiveness of communication is behind delivering high software performance in GDAD environment.

Fifthly, the research provides empirical support to the application of agile EA in measuring GDAD software performance. This is a significant contribution because the majority of the extant studies define EA from business (organisational) perspective only; this research instead views agile EA as the outcome of both technology and business capability-building process. Thus, the study represents a first attempt at using

agile EA from the technical side in conjunction with the business point of view to examine GDAD from a different angle. This coupling of the two core dimensions of agile EA provides a logical rationale to theorise the linkage between EA and agile development.

Sixthly, the model proposed in the research is an original contribution. It addresses the GDAD communication research gap identified by Alzoubi, Gill and Al-Ani (2016) and Hummel, Rosenkranz and Holten (2013) that there is a lack of confirmatory and explanatory studies about GDAD communication, especially when related to project characteristics issues such as using EA in GDAD. As such, the research fills this missing link and provides a comprehensive view on how agile EA enhances active communication and GDAD performance and how active performance enhances GDAD performance. This was done by evaluating and validating the research model using survey and case study methods. Therefore, this study has an important role in increasing the understanding of the GDAD from the communication perspective.

Seventhly, previous studies have indicated the need for GDAD communication related studies and stated that the research designs and studies are required to fit the current state of theory and research (Conboy 2009; Lee & Xia 2010). Accordingly, this study answered that call and complements Common Ground theory in GDAD environment. Hence, this study contributes by demonstrating that the Common Ground theory is relevant for understanding the relationship between agile EA and communication process. Although Common Ground theory has its origin in the fields of linguistics and cognitive psychology, our findings suggest that the theory has the potential to explain IS work contexts. Therefore, this study provides a starting point for research applying the Common Ground theory in studying agile software development.

Eighthly, the study adds to the current body of knowledge on agile EA implementation benefits. The majority of EA researches focus on post implementation benefits rather than the implementation issues.

Finally, in order to receive timely feedback from researchers' community, the findings of this research have been published and presented in the relevant peer-reviewed (highly

ranked) international academic conferences and scientific journals. This approach helps the researcher to proceed in the right direction during the different iterations of the research process. The outcomes of the study are publications in four high quality journals and two conferences. Two of the journal papers have already been published in Information and Management (an A* IS journal) and Journal of Software. Some of these papers were made freely accessible online to enable more scholars to get use of the benefits of these papers. A complete list of the research publications that have been delivered during this research project lifecycle is presented in the beginning of this thesis.

10.4.2 Contribution to Practice

GDAD organisation management is increasingly acknowledging the role of EA in the development process, in particular in large scale complex environments. EA can no longer be viewed as a supporting service for agile development. Instead, it has a leading role to play in both business and technology domains of GDAD (Gill 2015b). Likewise, enterprise architects are no longer seen as passive directors rather seen as partners who are actively involved in the GDAD project iterations. It is hoped that the outcomes of this study would contribute to both EA and GDAD practice in the context of achieving higher communication efficiency and communication effectiveness, and ultimately higher software performance. Therefore, the results of this study have several important implications for practitioners.

Firstly, the SLR results suggest that it is possible to achieve efficient and effective communication in GDAD, which facilitates software project performance. Moreover, this research provides evidence for practice suggesting that it is necessary to focus on human and organisational factors to increase GDAD communication efficiency and effectiveness. Specifically, GDAD communication requires a strong emphasis on personal communication and trust as well as a high level of tacit knowledge embedded in the team.

Secondly, traditional EA is perceived as a hindrance by agile teams and this study offers new insights for management by providing empirical evidence that GDAD

communication can be enhanced by using agile EA without burdening the agile teams and without unnecessarily increasing the number of visits between distributed sites or introducing additional communication tools or technology. Rather, it suggests that sharing agile integrated EA views (e.g. business architecture view, technology architecture view) among GDAD teams will serve as a base or a common language that will increase the common ground (common understanding) about the holistic business and technology landscape of an enterprise and related requirements. These shared views of agile EA will lead to more efficient and effective communication among teams working on different aspects or components of software projects in GDAD environment. Moreover, agile approaches in GDAD environment are not a silver bullet for optimum stakeholder coordination (Pikkarainen et al. 2008). Relying too much on tacit interpersonal knowledge and people willing to share it is also a risk, especially in GDAD environment (Boehm & Turner 2003), where informal communication is limited. Thus, the communication obstacles can be addressed by increasing the awareness of project requirements among GDAD teams using agile EA.

Thirdly, this study offers empirical evidence about the impact of agile EA on GDAD project performance. Unlike traditional EA approach that has been criticised for not delivering value, this study highlights the value of using agile EA to support high performance software projects. As discussed earlier, agile EA represents both business and technology domains, and should evolve as the GDAD projects are delivered in small iterations. In dynamic technology-enabled business environments, GDAD organisations may find it more difficult to align their business and technology views. Thus, GDAD organisations should integrate these two views in their daily operations. This research offers clear indications for GDAD practitioners that agile EA is an important factor in delivering high performance projects.

Fourthly, practitioners need to be aware of the multifaceted nature of GDAD active communication and understand the tension between its two different dimensions (e.g. efficiency and effectiveness) in order to build appropriate types of GDAD communication environment. For instance, when time and cost are top priorities, teams can be better off by selectively communicating (rather than too much or unnecessary communication) and thus increasing communication efficiency. Communication

channels such as instant messaging can be useful for improving response efficiency. GDAD teams can maintain an appropriate balance between the two active communication dimensions by using synchronous and asynchronous communication channels (according to the extent of efficiency and effectiveness), and implementing such agile development practices as daily meeting and weekly meetings. Moreover, this indicates to practitioners that agile principles could be fine-tuned to deliver high performance projects (Batra et al. 2010; Hummel et al. 2015).

Finally, the research findings emphasise the importance of actual employment of agile EA in GDAD environment in development iterations that involve responding to the business environmental changes and technical requirements changes. Hence, this research indicates that the management of GDAD organisations may consider paying more attention to agile EA such as training teams' members on working with agile EA as their supportive communication tool or enabler in their daily activities. Based on the study presented in this thesis, it can be suggested that agile EA could be considered as an integral part of the organisational communion processes but could also be part of the culture of GDAD organisations.

10.4.3 Contribution to Teaching and Learning

Employing agile EA as an enabler of GDAD communication has never been empirically investigated before (to the best of researcher's knowledge). So, investigating agile EA in the context of GDAD is a new learning experience. The findings of this study have some implications for the teaching and learning community. Research model and the findings of this study can be used as a base to train GDAD team members on the value of agile EA in enhancing communication and collaboration with other GDAD members. It can also be included in the curriculum to help students to understand the value of agile EA and its application to complex GDAD projects. Thus, this study may contribute in developing new or improving existing subjects that are taught in University of Technology Sydney such as Enterprise Software Architecture and Middleware (postgraduate degree), Enterprise Architecture Practice (undergraduate degree), Software Engineering Practice (undergraduate degree), and Applied Adaptive EA Pipeline (industry short course).

10.5 Limitations of the Study

GDAD faces many challenges such as reduced coordination, collaboration, group awareness, knowledge management, project and process management, risk management, and process support. These problems arise due to many reasons such as temporal differences, geographical differences, socio-cultural differences, and communication. This research addressed the communication issue through developing a research model and tested it using survey and semi-structured interviews data collection techniques. Despite the above-mentioned contributions, like any other studies, this study has also some limitations that should be considered when interpreting its findings. All steps were taken, to any extent possible, to mitigate or minimise the effects of these limitations. These limitations may mark the need for further research in the future.

Firstly, it was not an easy task to identify and develop the measures for the complex concepts in hand such as agile EA, GDAD communication, and software performance. Although we employed a rigorous, multiphase approach to the development of new measures for agile EA measurement development, the final measurement items may have some weaknesses. The identified concepts and measures reflected the most critical aspects of agile EA from the practitioners' point of view; however, they may not cover the whole range of the theoretical domain of the construct. Agile EA items included alignment between business and technical views, updating EA, defining the project by EA, defining the role of solution architecture that reflects EA and guides projects, and the governing role of EA in the GDAD project. However, this list may not be exhaustive because there are other potentially important EA components, which may be of interest to other researchers. This is typical of the nature of any research project as more concepts and measures can be identified to study other aspects of agile EA in this continuous process of research and development.

Secondly, each of the sub-constructs of on-time completion and on-budget completion consist of only two measurement items. They pass the minimum requirements for construct development according to Hair et al. (2010) who state that a construct should at least be measured by two indicators as well as the construct validity tests (see Ch.6).

However, these two sub-constructs are considered sufficient and fit for the scope of this study and hence, they are appropriate to address the research questions in hand.

Thirdly, one of the early concerns was the number of surveys that had been returned in the pilot survey; only 37 replies were obtained. Despite the comprehensive discussion which was provided about the adequacy and validity of this number for the specific purposes of the pilot survey, a higher number of responses would have nevertheless been more preferable. Higher number of responses would also have helped in excluding all possible bias in the results and enabled a more in-depth analysis of the data. The response rate of the full-scale survey of this study was proven to be statistically adequate with 160 responses, which eliminates the pilot study limitations. Nevertheless, a desirable goal was to obtain more responses that help to achieve additional confidence in the generalisability of the findings (Hair et al. 2010).

Finally, it is important to note here that there are a number of different definitions of EA and agile EA. Although a definition for agile EA was provided in the introduction section of the survey, the respondents may have used their own experience based thinking in interpreting or the misinterpreting measurements. This could be considered as a limitation of this study although we have not found any statistical evidence of potential misinterpretation to the measurement.

10.6 Prospective Directions for Future Research

Like any other research, this study needs to be viewed with its limitations, which are essentially the future research directions in this important and complex area of integrated agile EA and GDAD development. Firstly, the research model could be further tested by other research methods such as action research or case studies. In addition, the framework (Figure 3.2) itself could be used for further research to discover and examine other success factors and communication challenges.

Secondly, to the best of the researcher's knowledge, this is a first attempt to quantitatively measure the impact of agile EA on GDAD. Since sound measurements can only be established through a series of replications and validations, future research

may further validate and refine the measurement scales with comparative data. Furthermore, more concepts and measures can be identified to study other aspects of agile EA in this continuous process of research and development. This study fills a small gap in the research and does not claim to address all aspects of such large and complex domains of EA.

Thirdly, the scope of this study was to study the impact of agile EA as a whole on GDAD. This is because the individual GDAD projects or solution architectures include a subset of different integrated business and technology views of the agile EA. For instance, software is developed to address business concerns. Thus, it is not appropriate to have an artificial separation between business and technology views. Thus, the study of individual views of EA was not the aim of this research and is not practical, and is beyond the scope of this study. Other researchers may investigate individual domains or views (e.g. business and technology) of agile EA and their impact on GDAD project communication and performance. Moreover, future research may investigate the impact of agile EA on other aspects of the GDAD such as legal compliance, team habits and customer complaints.

Fourthly, while the results of this study revealed that communication efficiency positively impacts communication effectiveness, future research may investigate if the reverse causality can happen under certain conditions. In addition, although the study findings revealed that communication efficiency and communication effectiveness affect differently the project performance in GDAD environment, these subtle yet significant relationships should not be overlooked, and future research may investigate their impacts on GDAD performance from other perspectives.

Fifthly, due to the fact that theories have not been commonly used in the research on GDAD communication, the application of Common Ground theory in further research on agile development communication would be valuable. Furthermore, to obtain a deeper understanding of the communication issue in GDAD environment, GDAD communication could be investigated from the perspectives of other related theories such as Activity Theory (Engeström 1999) and Coordination Theory (Malone &

Crowston 1994), which would increase the knowledge about agile EA, GDAD communication and their impacts on GDAD performance.

Sixthly, one possible solution of the small response number is to collect quantitative data during a longer period in future research. Moreover, future research may employ additional surveying techniques such as the traditional mail survey approach that may potentially increase the overall response rate.

Finally, future research may employ a survey design that selects separate respondents for the independent and dependent variables of the research model in order to reduce the potential for common method bias. Also, more explanation to participants about agile EA and the differences with traditional EA may be needed in future research.

10.7 Concluding Remarks

ASD is a progressive technique to mitigate the challenges of changeable customer requirements, while still meeting schedule, cost, functionality, and quality standards. There is, however, very limited empirical research on the viability of GDAD that uses a blend of efficient and effective communication and measures impacts throughout the agile development iterations. As GDAD trend continues growing for the purposes of cost-saving development, GDAD organisations must understand how to maximise communication efficiency and communication effectiveness among GDAD teams. This challenge affects directly how organisations understand, build, and deliver performance standards. Also, this challenge affects how organisations develop collaboration tools and how teams share information in GDAD environment.

This research, through a process of theory-based model development and rigorous empirical investigation of the proposed model, is an attempt to bridge the above mentioned important research gap and provides an empirical evidence to mitigate the impact of this challenge and understand its impacts on software project performance in GDAD environment. That is, this research empirically revealed that using agile EA vision among teams enhances both communication efficiency and communication effectiveness, and enhances software performance in GDAD environment. Moreover,

this research has provided empirical evidence of the positive impacts of communication efficiency and communication effectiveness on GDAD software project performance.

Therefore, in a nutshell, it can be concluded that agile EA may help GDAD teams to translate synchronous communication requirements into efficient and effective levels of asynchronous communication. Initially, this may be a difficult task, but once achieved, GDAD teams may steadily achieve successes similar to that achieved by co-located ASD teams.

10.8 Chapter Summary

This chapter discussed the contributions and limitations of this study, and the future research opportunities arising from this study. As a theoretical contribution, this study rigorously, through empirical investigation using survey and semi-structured interviews, showed positive impacts of agile EA on active communication and software performance, and a positive impact of active communication on software performance in GDAD environment. As a practical contribution, this study calls for paying more attention to the benefits of using agile EA by practitioners which impacts positively on the performance of a GDAD software project. As this study is among the first studies that investigate the role of agile EA in GDAD environment, more research is required to validate its findings. Table 10.1 summarises the outcomes of this study according to the research questions and suggestions for further research in the context of agile EA driven GDAD communication.

Table 10.1: Outcomes of the study

Research Question	Method	Study Outcome	Future Research
Main question: *How to enhance communication in geographically distributed agile development?*	PhD	Thesis	• The framework (Figure 3.2) could be used for further research to examine other success factors (assess other communication challenges) such as organisational factors and human factors • This study investigated the role of agile EA as a whole on GDAD communication; future research may investigate individual domains or views (e.g. business and technology) of agile EA and their impact on GDAD project communication and performance. Moreover, future research may investigate the impact of agile EA on other aspects of the GDAD such as legal compliance, team habits and customer complaints • Future research should further investigate the relationship between the two active communication dimensions (communication efficiency and communication effectiveness) • This study used Common Ground theory to investigate the role of GDAD communication. Further research may use other related theories such as Activity Theory and Coordination Theory
RQ1: *What are the challenges or factors that limit GDAD communication?* ***RQ2:*** *Which strategies, techniques or practices have been used to overcome these challenges and enhance GDAD communication?*	SLR	• Alzoubi et al. 2016 • Alzoubi & Gill 2014	
RQ3: *How does agile EA affect GDAD active communication?* ***RQ4:*** *How does agile EA affect GDAD software performance?* ***RQ5:*** *How does GDAD active communication affect GDAD software performance?* ***RQ6:*** *How can the findings of this research assist GDAD practitioners in overcoming GDAD communication issues?*	Literature review and expert's assessment Literature review Survey and case studies	• Alzoubi & Gill 2016 • Alzoubi & Gill 2015 • Alzoubi et al. 2016 • Alzoubi & Gill 2017a (forthcoming) • Alzoubi & Gill 2017b (forthcoming)	

Bibliography

Abrahamsson, P., Conboy, K. & Wang, X. 2009, '"Lots done, more to do": the current state of agile systems development research', *European Journal of Information Systems*, vol. 18, pp. 281–4.

Abrahamsson, P., Salo, O., Ronkainen, J. & Warsta, J. 2002, 'Agile software development methods: review and analysis', Technical report, VTT Publications, Finland.

Agerfalk, P.J., Fitzgerald, B., Holmstrom, H., Lings, B., Lundell, B. & Conchúir, E.O. 2005, 'A framework for considering opportunities and threats in distributed software development', *International Workshop on Distributed Software Development*, Citeseer, pp. 47-61.

Agerfalk, P., Fitzgerald, B. & Slaughter, S. 2009, 'Flexible and distributed information systems development: state of the art and research challenges', *Information Systems Research*, vol. 20, no. 3, pp. 317-28.

Agrawal, M. & Chari, K. 2007, 'Software effort, quality, and cycle time: a study of CMM level 5 projects', *IEEE Transactions on Software Engineering*, vol. 33, no. 3, pp. 145-56.

Agile Manifesto 2001, 'Manifesto for agile software development', viewed 20 September 2013, <http://www.agilemanifesto.org/>.

Akhigbe, O.S. 2014, 'Business intelligence-enabled adaptive enterprise architecture', Master thesis, University of Ottawa.

Aladwani, A.M. 2002, 'An integrated performance model information systems projects', *Journal of Management Information Systems*, vol. 19, no. 1, pp. 185-210.

Ali Babar, M., Ihme, T. & Pikkarainen, M. 2009, 'An industrial case of exploiting product line architectures in agile software development', *In Proceedings of the 13th International Software Product Line Conference*, Carnegie Mellon University, pp. 171-9.

Al-Kilidar, H., Cox, K. & Kitchenham, B. 2005, 'The use and usefulness of the ISO/IEC 9126 quality standard', *International Symposium on Empirical Software Engineering*, IEEE, pp. 126-32.

Aloudat, A. 2010, 'Location-based mobile phone service utilisation for emergency management in Australia', PhD thesis, University of Wollongong.

Alzoubi, Y.I. & Gill, A.Q. 2014, 'Agile global software development communication challenges: a systematic review', *In Proceedings of the 18th Pacific Asia Conference on Information Systems (PACIS 2014)*, p. 20.

Alzoubi, Y.I. & Gill, A. 2015, 'An agile enterprise architecture driven model for geographically distributed agile development', *In Proceedings of the 24th International Conference on Information System Development (ISD 2015)*, Harbin, China.

Alzoubi, Y.I., Gill, A.Q. & Al-Ani, A. 2015, 'Distributed agile development communication: an agile architecture driven framework', *Journal of Software*, vol. 10, no. 6, pp. 681-94.

Alzoubi, Y.I. & Gill, A.Q. 2016, 'An agile enterprise architecture-driven model for geographically distributed agile development', in D. Vogel, X. Guo, H. Linger, C. Barry, M. Lang & C. Schneider (eds), *Transforming Healthcare Through Information Systems*, Springer International Publishing, pp. 63-77.

Alzoubi, Y.I., Gill, A.Q. & Al-Ani, A. 2016, 'Empirical studies of geographically distributed agile development communication challenges: a systematic review', *Information and Management*, vol. 53, no. 1, pp. 22-37.

Ambler, S. 2006, 'Crossing the chase', viewed 27 December 2014, <http://www.drdobbs.com/architecture-anddesign/187200223;jsessionid=SS2EHYZRLEVHTQE1GHPSKHWATMY32JVN?pgno=2.>.

Ambler, S. 2011, 'Agile and enterprise architecture ', viewed 25 December 2014, <http://www.ambysoft.com/surveys/agileEA201103.html>.

Ambler, S. 2012a, 'Agility at scale survey: results from the summer 2012 DDJ State of the IT Union Survey', viewed 4 June 2016, <http://www.ambysoft.com/surveys/stateOfITUnion201209.html>.

Ambler, S. 2012b, 'Agile success factors', Dr.Dobb's, The World of Software Development, viewed 4 June 2016, <http://www.drdobbs.com/ architechture-and-design/agilie-success-factors/232601858>.

Ambler, S. 2014a, 'Agile modeling: communication on agile software projects', viewed 20 September 2014, <http://www.agilemodeling.com/essays/communication.htm>.

Ambler, S. 2014b, 'Agile data: agile enterprise architecture', viewed 25 January 2014, <http://agiledata.org/essays/enterpriseArchitecture.html>.

Ambler, S. 2015, 'Disciplined agile 2.0: geographically distributed agile teams', viewed 27 December 2015, <http://www.disciplinedagiledelivery.com/agility-at-scale/geographically-distributed-agile-teams/>.

Anderson, D. 2010, *Kanban: successful evolutionary change for your technology business*, Blue Hole Press.

Avison, D. & Fitzgerald, G. 2003, *Information systems development: methodologies, techniques and tools*, 3 edn, McGraw Hill, Maidenhead, UK.

Avritzer, A., Paulish, D., Cai, Y. & Sethi, K. 2010, 'Coordination implications of software architecture in a global software development project', *Journal of Systems and Software*, vol. 83, no. 10, pp. 1881-95.

Babbie, E. 2001, *The practice of social research*, 9 edn, Wadsworth Thomson Learning, Belmont, California.

Baker, T.L. & Risley, A.J. 1994, *Doing social research*, McGraw-Hill, New York.

Balijepally, V., Mahapatra, R., Nerur, S. & Price, K.H. 2009, 'Are two heads better than one for software development? The productivity paradox of pair programming', *MIS Quarterly*, pp. 91-118.

Bannerman, P.L., Hossain, E. & Jeffery, R. 2012, 'Scrum practice mitigation of global software development coordination challenges: a distinctive advantage?', *In Proceedings of the 45th Hawaii International Conference on System Science (HICSS)*, IEEE, pp. 5309-18.

Barclay, D., Higgins, C. & Thompson, R. 1995, 'The partial least squares (PLS) approach to causal modeling: personal computer adoption and use as an illustration', *Technology Studies*, vol. 2, no. 2, pp. 285-309.

Barlow, J.B., Giboney, J., Keith, M.J., Wilson, D., Schuetzler, R., Lowry, P.B. & Vance, A. 2011, 'Overview and guidance on agile development in large organizations', *Communications of the Association for Information Systems*, vol. 29, no. 2, pp. 25-44.

Baron, R.M. & Kenny, D.A. 1986, 'The moderator–mediator variable distinction in social psychological research: conceptual, strategic, and statistical considerations', *Journal of Personality and Social Psychology*, vol. 51, no. 6, pp. 1173-82.

Bass, L., Clements, P. & Kazman, R. 2013, *Software architecture in practice*, 3 edn, Addison-Wesley, Upper Saddle River, NJ.

Bass, L. & Kazman, R. 1999, *Architecture-based development (Report No. CMU/SEI-99-TR-007, ESC-TR-99-007)*, Carnegie Mellon Software Engineering Institute.

Batra, D. 2009, 'Modified agile practices for outsourced software projects', *Communications of the ACM*, vol. 52, no. 9, pp. 143-8.

Batra, D., VanderMeer, D. & Dutta, K. 2011, 'Extending agile principles to larger, dynamic software projects: a theoretical assessment', *Journal of Database Management (JDM)*, vol. 22, no. 4, pp. 73-92.

Batra, D., Xia, W., VanderMeer, D. & Dutta, K. 2010, 'Balancing agile and structured development approaches to successfully manage large distributed software projects: a case study from the cruise line industry', *Communications of the Association for Information Systems*, vol. 27, no. 1, pp. 379-94.

Beck, K. 2000, *Extreme programming explained*, Addison-Wesley Pearson Education, Boston.

Beck, K. & Andres, C. 2005, *Extreme programming explained: embrace change*, Addison-Wesley, Boston.

Beck, P., Jiang, J.J. & Klein, G. 2006, 'Prototyping mediators to project performance: learning and interaction', *Journal of Systems and Software*, vol. 79, no. 7, pp. 1025-35.

Becker, J.-M., Klein, K. & Wetzels, M. 2012, 'Hierarchical latent variable models in PLS-SEM: guidelines for using reflective-formative type models', *Long Range Planning*, vol. 45, no. 5, pp. 359-94.

Becker, J. & Niehaves, B. 2007, 'Epistemological perspectives on IS research: a framework for analysing and systematizing epistemological assumptions', *Information Systems Journal*, vol. 17, no. 2, pp. 197-214.

Benbasat, I., Goldstein, D.K. & Mead, M. 1987, 'The case research strategy in studies of information systems', *MIS Quarterly*, pp. 369-86.

Berczuk, S. 2007, 'Back to basics: the role of agile principles in success with an distributed Scrum team', *Agile Conference (AGILE)*, IEEE, pp. 382-8.

Bernard, S.A. 2012, *An introduction to enterprise architecture*, 3 edn, AuthorHouse, Bloomington, IN.

Bernard, H.R. & Bernard, H.R. 2012, *Social research methods: qualitative and quantitative approaches*, Sage Publications.

Bhalerao, S. & Ingle, M. 2010, 'Analyzing the modes of communication in agile practices', *In Proceedings of the 3rd International Conference on Computer Science and Information Technology (ICCSIT)*, vol. 3, IEEE, pp. 391-5.

Boden, A., Nett, B. & Wulf, V. 2007, 'Coordination practices in distributed software development of small enterprises', *In Proceedings of the Second International Conference on Global Software Engineering (ICGSE)*, IEEE, pp. 235-46.

Boehm, B.W. 1988, 'A spiral model of software development and enhancement', *Computer*, vol. 21, no. 5, pp. 61-72.

Boehm, B. & Turner, R. 2003, *Balancing agility and discipline: a guide for the perplexed*, Addison-Wesley Professional.

Boh, W.F. & Yellin, D. 2006, 'Using enterprise architecture standards in managing information technology', *Journal of Management Information Systems*, vol. 23, no. 3, pp. 163-207.

Bosch, J. & Bosch-Sijtsema, P. 2010, 'Coordination between global agile teams: from process to architecture', in D. Šmite et al. (eds), *Agility Across Time and Space*, Springer Berlin Heidelberg, pp. 217-33.

Bostrom, R.P. 1989, 'Successful application of communication techniques to improve the systems development process', *Information and Management*, vol. 16, no. 5, pp. 279-95.

Brennan, S.E. 1998, 'The grounding problem in conversations with and through computers', in S.R. Fussel & R.J. Kreuz (eds), *Social and cognitive approaches to interpersonal communication*, Psychology Press, Taylor & Francis Group, New York, pp. 210-25.

Buckl, S., Matthes, F., Monahov, I., Roth, S., Schulz, C. & Schweda, C.M. 2011, 'Towards an agile design of the enterprise architecture management function', *In Proceedings of the 15th International Enterprise Distributed Object Computing Conference Workshops (EDOCW)*, IEEE, pp. 322-9.

Burstein, F. & Gregor, S. 1999, 'The systems development or engineering approach to research in information systems: An action research perspective', *In Proceedings of the 10th Australasian Conference on Information Systems*, Victoria University of Wellington, New Zealand, pp. 122-34.

Bush, A.A., Tiwana, A. & Tsuji, H. 2008, 'An empirical investigation of the drivers of software outsourcing decisions in Japanese organizations', *Information and Software Technology*, vol. 50, no. 6, pp. 499-510.

Calefato, F., Damian, D. & Lanubile, F. 2012, 'Computer-mediated communication to support distributed requirements elicitations and negotiations tasks', *Empirical Software Engineering*, vol. 17, no. 6, pp. 640-74.

Cao, L., Mohan, K., Xu, P. & Ramesh, B. 2009, 'A framework for adapting agile development methodologies', *European Journal of Information Systems*, vol. 18, no. 4, pp. 332-43.

Carlile, P.R. 2004, 'Transferring, translating, and transforming: an integrative framework for managing knowledge across boundaries', *Organization Science*, vol. 15, no. 5, pp. 555-68.

Carmel, E. & Abbott, P. 2007, 'Why 'nearshore' means that distance matters', *Communications of the ACM*, vol. 50, no. 10, pp. 40-6.

Carmel, E., Espinosa, J.A. & Dubinsky, Y. 2010, 'Follow the Sun: workflow in global software development', *Journal of Management Information Systems*, vol. 27, no. 1, pp. 17-38.

Cataldo, M. & Ehrlich, K. 2011, 'The impact of the structure of communication patterns in global software development: an empirical analysis of a project using agile methods', *Institute for Software Research, Carnegie Mellon University*.

Cataldo, M. & Herbsleb, J.D. 2008, 'Communication patterns in geographically distributed software development and engineers' contributions to the development effort', *In the Proceedings of the 2008 International Workshop on Cooperative and Human Aspects of Software Engineering*, ACM, pp. 25-8.

Cavana, R., Delahaye, B.L. & Sekeran, U. 2001, *Applied business research: qualitative and quantitative methods*, John Wiley & Sons, Australia.

Cavaye, A.L. 1996, 'Case study research: a multi-faceted research approach for IS', *Information Systems Journal*, vol. 6, no. 3, pp. 227-42.

Cecez-Kecmanovic, D. 2007, 'Critical research in information systems: the question of methodology', *In Proceedings of the ECIS*, pp. 1446-57.

Cenfetelli, R.T. & Bassellier, G. 2009, 'Interpretation of formative measurement in information systems research', *MIS Quarterly*, vol. 33, no. 4, pp. 689-707.

Chen, W. & Hirschheim, R. 2004, 'A paradigmatic and methodological examination of information systems research from 1991 to 2001', *Information Systems Journal*, vol. 14, no. 3, pp. 197-235.

Chin, W.W. 1998, 'Commentary: issues and opinion on structural equation modeling', *MIS Quarterly*, vol. 22, pp. vii–xvi.

Chin, W.W. 2010, 'How to write up and report PLS analyses', in V.E. Vinzi, W.W. Chin, J. Henseler & H. Wang (eds), *Handbook of partial least squares, Springer Handbooks of Computational Statistics*, Springer Berlin Heidelberg, pp. 655-90.

Chin, W.W. & Todd, P.A. 1995, 'On the use, usefulness, and ease of use of structural equation modeling in MIS research: a note of caution', *MIS Quarterly*, vol. 19, pp. 237-46.

Chow, T. & Cao, D.-B. 2008, 'A survey study of critical success factors in agile software projects', *Journal of Systems and Software*, vol. 81, no. 6, pp. 961-71.

CIO Council 2001, 'A practical guide to federal enterprise architecture', <https://cio.gov>.

Clark, H.H. 1996, *Using Language*, Cambridge University Press, Cambridge.

Clark, H.H. & Brennan, S.E. 1991, 'Grounding in communication', *Perspectives on Socially Shared Cognition*, vol. 13, no. 1991, pp. 127-49.

Cockburn, A. 2002, *Agile software development*, Addison-Wesley, Boston.

Cockburn, A. 2004, *Crystal clear: a human-powered methodology for small teams*, Pearson Education.

Cockburn, A. 2007, *Agile software development: the cooperative game*, Addison-Wesley, Harlow.

Cohen, J. 1988, *Statistical Power Analysis for the Behavioral Sciences*, 2nd edn, Lawrence Erlbaum Associates, Hillsdale, New Jersey.

Conboy, K. 2009, 'Agility from first principles: reconstructing the concept of agility in information systems development', *Information Systems Research*, vol. 20, no. 3, pp. 329-54.

Conboy, K. & Fitzgerald, B. 2004, 'Toward a conceptual framework of agile methods', in C. Zannier, H. Erdogmus & L. Lindstrom (eds), *Extreme Programming and Agile Methods-XP/Agile Universe 2004*, Springer Berlin Heidelberg, pp. 105-16.

Conboy, K., Coyle, S., Wang, X. & Pikkarainen, M. 2010, 'People over process: key people challenges in agile development', *IEEE Software*, vol. 28, no. 4, pp. 48–57.

Conchúir, E.Ó., Ågerfalk, P.J., Olsson, H.H. & Fitzgerald, B. 2009, 'Global software development: where are the benefits?' *Communications of the ACM*, vol. 52, no. 8, pp. 127-31.

Cramton, C.D. 2002, 'Finding common ground in dispersed collaboration', *Organizational Dynamics*, vol. 30, no. 4, pp. 356-67.

Creswell, J.W. 2009, *Research design: qualitative, quantitative, and mixed methods approaches*, 2nd edn, SAGE Publications, Incorporated.

Cummings, J.N. 2004, 'Work groups, structural diversity, and knowledge sharing in a global organization', *Management Science*, vol. 50, no. 3, pp. 352-64.

Dabbish, L., Stuart, C., Tsay, J. & Herbsleb, J. 2012, 'Social coding in GitHub: transparency and collaboration in an open software repository', *In Proceedings of the ACM 2012 conference on Computer Supported Cooperative Work*, ACM, pp. 1277-86.

Daft, R.L., Lengel, R.H. & Trevino, L.K. 1987, 'Message equivocality, media selection, and manager performance: implications for information systems', *MIS Quarterly*, pp. 355-66.

da Silva, F.Q., Costa, C., França, A.C.C. & Prikladinicki, R. 2010, 'Challenges and solutions in distributed software development project management: a systematic literature review', *In Proceedings of the 5th International Conference on Global Software Engineering (ICGSE)*, IEEE, pp. 87-96.

Dennis, A.R., Fuller, R.M. & Valacich, J.S. 2008, 'Media, tasks, and communication processes: a theory of media synchronicity', *MIS Quarterly*, vol. 32, no. 3, pp. 575-600.

Department of Defense, U.S. 2010, 'The DoDAF architecture framework version 2.02', viewed 7 December 2013, <http://dodcio.defense.gov/dodaf20.aspx>.

DeVito, J.A. 1988, *Human communication*, New York, NY: Harper and Row, Inc.

Dingsøyr, T., Fægri, T.E. & Itkonen, J. 2014, 'What is large in large-scale? A taxonomy of scale for agile software development', in A. Jedlitschka et al. (eds), *Product-Focused Software Process Improvement*, Springer International Publishing Switzerland, pp. 273-6.

Dingsøyr, T., Nerur, S., Balijepally, V. & Moe, N.B. 2012, 'A decade of agile methodologies: towards explaining agile software development', *Journal of Systems and Software*, vol. 85, no. 6, pp. 1213-21.

Di Vitantonio, G., Legh-Smith, J., Millar, W. & Wilkinson, M. 2006, 'Meeting business objectives through adaptive information and communications technology', *BT Technology Journal*, vol. 24, no. 4, pp. 113-20.

Dobson, P.J. 2001, 'The philosophy of critical realism—an opportunity for information systems research', *Information Systems Frontiers*, vol. 3, no. 2, pp. 199-210.

Dorairaj, S., Noble, J. & Malik, P. 2011, 'Effective communication in distributed agile software development teams', in A. Sillitti et al. (eds.), *Agile Processes in Software Engineering and Extreme Programming*, Springer Berlin Heidelberg, pp. 102-16.

Drury-Grogan, M.L. 2014, 'Performance on agile teams: relating iteration objectives and critical decisions to project management success factors', *Information and Software Technology*, vol. 56, no. 5, pp. 506-15.

DSDM 2013, 'DSDM consortium, dynamic systems development method Ltd', <http://www.dsdm.org/>.

Dubé, L. & Paré, G. 2003, 'Rigor in information systems positivist case research: current practices, trends, and recommendations', *MIS Quarterly*, pp. 597-636.

Dyba, T., Arisholm, E., Sjoberg, D.I., Hannay, J.E. & Shull, F. 2007, 'Are two heads better than one? On the effectiveness of pair programming', *IEE Software*, vol. 24, no. 6, pp. 12-5.

Dybå, T. & Dingsøyr, T. 2008, 'Empirical studies of agile software development: a systematic review', *Information and Software Technology*, vol. 50, no. 9, pp. 833-59.

Edwards, C. 2006, 'Agile Enterprise Architecture, Part 1', *USA: Process Wave*.

Eisenhardt, K.M. 1989, 'Building theories from case study research', *Academy of Management Review*, vol. 14, no. 4, pp. 532-50.

Engardio, P., Bernstein, A., Kripalani, M., Balfour, F., Grow, B. & Green, J. 2003, 'Is your job next?' *Business Week*, vol. 3, pp. 50-60.

Engeström, Y. 1999, 'Activity theory and individual and social transformation', *Perspectives on activity theory (19-38). Cambridge: University Press*.

Estler, H., Nordio, M., Furia, C.A., Meyer, B. & Schneider, J. 2012, 'Agile vs. structured distributed software development: a case study', *Seventh International Conference on Global Software Engineering (ICGSE)*, IEEE, pp. 11-20.

Estler, H.-C., Nordio, M., Furia, C.A., Meyer, B. & Schneider, J. 2014, 'Agile vs. structured distributed software development: a case study', *Empirical Software Engineering*, vol. 19, no. 5, pp. 1197-224.

European Union Commission 2003, 'Commission recommendation of 6 May 2003 concerning the definition of micro, small and medium-sized enterprises', *Official Journal of the European Union*, vol. 46, pp. 36-41.

Faul, F., Erdfelder, E., Lang, A.-G. & Buchner, A. 2007, 'G* Power 3: a flexible statistical power analysis program for the social, behavioral, and biomedical sciences', *Behavior Research Methods*, vol. 39, no. 2, pp. 175-91.

Fitzgerald, B., Hartnett, G. & Conboy, K. 2006, 'Customising agile methods to software practices at Intel Shannon', *European Journal of Information Systems*, vol. 15, no. 2, pp. 200-13.

Fornell, C. & Larcker, D.F. 1981, 'Structural equation models with unobservable variables and measurement error: algebra and statistics', *Journal of Marketing Research*, pp. 382-8.

Franke, U., Ekstedt, M., Lagerström, R., Saat, J. & Winter, R. 2010, 'Trends in enterprise architecture practice–a survey', in E. Proper, M.M. Lankhorst, M. Schönherr, J. Barjis & S. Overbeek (eds), *Trends in Enterprise Architecture Research*, Springer Berlin Heidelberg, pp. 16-29.

Froehlich, J. & Dourish, P. 2004, 'Unifying artifacts and activities in a visual tool for distributed software development teams', *In Proceedings of the 26th International Conference on Software Engineering*, IEEE Computer Society, pp. 387-96.

Frøkjær, E., Hertzum, M. & Hornbæk, K. 2000, 'Measuring usability: are effectiveness, efficiency, and satisfaction really correlated?' *In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ACM, pp. 345-52.

Fruhling, A. & De Vreede, G.-J. 2006, 'Field experiences with eXtreme programming: developing an emergency response system', *Journal of Management Information Systems*, vol. 22, no. 4, pp. 39-68.

Gable, G.G. 1994, 'Integrating case study and survey research methods: an example in information systems', *European Journal of Information Systems*, vol. 3, no. 2, pp. 112-26.

Gartner 2008, 'Key Issues for Enterprise Architecture', viewed 22 May 2016, <http://www.btmg.biz/EA_Kitty3/Documents/Gartner/Gartner6.pdf>.

Gaur, A.S. & Gaur, S.S. 2006, *Statistical methods for practice and research: a guide to data analysis using SPSS*, Sage Publications, New Delhi.

Gebauer, J., Tang, Y. & Baimai, C. 2008, 'User requirements of mobile technology: results from a content analysis of user reviews', *Information Systems and e-Business Management*, vol. 6, no. 4, pp. 361-84.

Gefen, D., Straub, D. & Boudreau, M.-C. 2000, 'Structural equation modeling and regression: guidelines for research practice', *Communications of the Association for Information Systems*, vol. 4, no. 1, pp. 1-76.

Gefen, D., Straub, D.W. & Rigdon, E.E. 2011, 'An update and extension to SEM guidelines for admnistrative and social science research', *MIS Quarterly*, vol. 35, no. 2, pp. iii-xiv.

Geisser, S. 1975, 'The predictive sample reuse method with applications', *Journal of the American Statistical Association*, vol. 70, no. 350, pp. 320-8.

Gill, A.Q. 2012, 'The gill framework: adaptive enterprise architecture toolkit', *ISBN-13: 978-1481276870*.

Gill, A.Q. 2013a, 'Towards the development of an adaptive enterprise service system model', *In Proceedings of the Nineteenth Americas Conference on Information Systems*, Chicago, Illinois, pp. 1-9.

Gill, A. 2013b, 'Defining a social architecture within the enterprise architecture context', White Paper, <www.orbussoftware.com/community>.

Gill, A.Q. 2014a, 'The Gill Framework - creating agile or adaptive enterprise project management capability for enterprise agility', *Asif Q Gill*, weblog, viewed 5 October 2014, <http://aqgill.blogspot.com.au/>.

Gill, A.Q. 2014b, 'The Gill Framework®', *Asif Q Gill*, weblog, viewed 5 October 2014, <http://www.aqgill.com/adoms/>.

Gill, A.Q. 2014c, 'Applying agility and living service systems thinking to enterprise architecture', *International Journal of Intelligent Information Technologies*, vol. 10, no. 1, pp. 1-15.

Gill, A.Q. 2014d, 'Hybrid adaptive software development capability: an empirical study', *Journal of Software*, vol. 9, no. 10, pp. 2614-21.

Gill, A.Q. 2015a, 'Distributed agile development: applying a coverage analysis approach to the evaluation of a communication technology assessment tool', *International Journal of e-Collaboration (IJeC)*, vol. 11, no. 1, pp. 57-76.

Gill, A.Q. 2015b, *Adaptive cloud enterprise architecture*, World Scientific Publishing Co. Pte. Ltd.

Gill, A.Q. & Bunker, D. 2011, 'Conceptualization of a context aware cloud adaptation (CACA) framework', *In Proceedings of the Ninth International Conference on Dependable, Autonomic and Secure Computing (DASC)*, IEEE, pp. 760-7.

Gill, A.Q. & Bunker, D. 2013, 'Towards the development of a cloud-based communication technologies assessment tool: an analysis of practitioners' perspectives', *VINE*, vol. 43, no. 1, pp. 57-77.

Gill, A.Q., Bunker, D. & Seltsikas, P. 2012, 'Evaluating a communication technology assessment tool (Ctat): a case of a cloud based communication tool', *In Proceedings of the PACIS 2012, paper 88*.

Glaser, B.G. 1978, *Theoretical sensitivity: advances in the methodology of grounded theory*, vol. 2, Sociology Press-Mill Valley, CA.

Gokpinar, B., Hopp, W.J. & Iravani, S.M. 2010, 'The impact of misalignment of organizational structure and product architecture on quality in complex product development', *Management Science*, vol. 56, no. 3, pp. 468-84.

Gold, A.H., Malhotra, A. & Segars, A.H. 2001, 'Knowledge management: an organizational capabilities perspective', *Journal of Management Information Systems*, vol. 18, no. 1, pp. 185-214.

Goles, T. & Hirschheim, R. 2000, 'The paradigm is dead, the paradigm is dead… long live the paradigm: the legacy of Burrell and Morgan', *Omega*, vol. 28, no. 3, pp. 249-68.

Gotz, O., Liehr-Gobbers, K. & Krafft, M. 2010, 'Evaluation of structural equation models using the partial least squares (PLS) approach', in V.E. Vinzi, W. W. Chin, J. Henseler & H. Wang (eds), *Handbook of partial least squares*, Springer Berlin Heidelberg, pp. 691-711.

Green, R., Mazzuchi, T. & Sarkani, S. 2010, 'Communication and quality in distributed agile development: an empirical case study', *In Proceedings of the World Academy of Science, Engineering and Technology*, vol. 61, pp. 322-8.

Gregor, S. 2006, 'The nature of theory in information systems', *MIS Quarterly*, pp. 611-42.

Gregor, S., Hart, D. & Martin, N. 2007, 'Enterprise architectures: enablers of business strategy and IS/IT alignment in government', *Information Technology and People*, vol. 20, no. 2, pp. 96-120.

Gregor, S. & Hevner, A.R. 2013, 'Positioning and presenting design science research for maximum impact', *MIS Quarterly*, vol. 37, no. 2.

Gregor, S. & Klein, G. 2014, 'Eight obstacles to overcome in the theory testing genre', *Journal of the Association for Information Systems*, vol. 15, no. 11, pp. i-xix.

Guba, E.G. & Lincoln, Y.S. 1994, 'Competing paradigms in qualitative research', *Handbook of Qualitative Research*, vol. 2, no. 163-194.

Hair, J.F., Black, W.C., Babin, B.J. & Anderson, R.E. 2010, *Multivariate data analysis*, 7 edn, Pearson Prentice Hall, New York.

Hair, J., Hult, T., Ringle, C. & Sarstedt, M. 2014, *A primer on partial least squares structural equation modeling (PLS-SEM)*, 1 edn, Sage Publications.

Hair, J.F., Ringle, C.M. & Sarstedt, M. 2011, 'PLS-SEM: indeed a silver bullet', *Journal of Marketing Theory and Practice*, vol. 19, no. 2, pp. 139-52.

Hammad, M., Hammad, M., Bani-Salameh, H. & Fayyoumi, E. 2014, 'Measuring developers' design contributions in evolved software projects', *Journal of Software*, vol. 9, no. 12, pp. 3005-11.

Heiberg, S., Puus, U., Salumaa, P. & Seeba, A. 2003, 'Pair-programming effect on developers productivity', in M. Marchesi & G. Succi (eds), *In Extreme Programming and Agile Processes in Software Engineering*, Springer Berlin Heidelberg, pp. 215-24.

Helquist, J.H., Deokar, A., Meservy, T. & Kruse, J. 2011, 'Dynamic collaboration: participant-driven agile processes for complex tasks', *ACM SIGMIS Database*, vol. 42, no. 2, pp. 95-115.

Henseler, J., Ringle, C.M. & Sarstedt, M. 2012, 'Using partial least squares path modeling in advertising research: basic concepts and recent issues', in S. Okazaki & E. Cheltenham (eds), *Handbook of research on international advertising*, Elgar Publishing, pp. 252-76.

Henseler, J., Ringle, C.M. & Sarstedt, M. 2015, 'A new criterion for assessing discriminant validity in variance-based structural equation modeling', *Journal of the Academy of Marketing Science*, pp. 1-21.

Henseler, J., Ringle, C.M. & Sinkovics, R.R. 2009, 'The use of partial least squares path modeling in international marketing', *Advances in International Marketing (AIM)*, vol. 20, pp. 277-319.

Henseler, J. & Sarstedt, M. 2013, 'Goodness-of-fit indices for partial least squares path modeling', *Computational Statistics*, vol. 28, no. 2, pp. 565-80.

Herbsleb, J.D. 2007, 'Global software engineering: the future of socio-technical coordination', *Future of Software Engineering*, IEEE Computer Society, pp. 188-98.

Herbsleb, J.D. & Mockus, A. 2003, 'An empirical study of speed and communication in globally distributed software development', *IEEE Transactions on Software Engineering*, vol. 29, no. 6, pp. 481-94.

Herbsleb, J.D. & Moitra, D. 2001, 'Global software development', *IEEE Software*, vol. 18, no. 2, pp. 16-20.

Highsmith, J.A.I. 2000, *Adaptive software development: a collaborative approach to managing complex systems*, Dorset House Publishing, New York.

Highsmith, J. 2002, *Agile software development ecosystems*, Pearson Education, Boston, MA.

Highsmith, J. 2009, *Agile project management: creating innovative products*, Pearson Education, USA.

Highsmith, J. & Cockburn, A. 2001, 'Agile software development: the business of innovation', *IEEE Computer Society*, vol. 34, no. 9, pp. 120-7.

Hinds, P.J. & Mortensen, M. 2005, 'Understanding conflict in geographically distributed teams: the moderating effects of shared identity, shared context, and spontaneous communication', *Organization Science*, vol. 16, no. 3, pp. 290-307.

Hirschheim, R. 1985, 'Information systems epistemology: an historical perspective', *Research Methods in Information Systems*, pp. 13-35.

Hirschheim, R. & Klein, H.K. 1989, 'Four paradigms of information systems development', *Communications of the ACM*, vol. 32, no. 10, pp. 1199-216.

Hoda, R., Noble, J. & Marshall, S. 2010, 'How much is just enough?: some documentation patterns on Agile projects', *In Proceedings of the 15th European Conference on Pattern Languages of Programs*, ACM, p. 13.

Hofstede, G., Hofstede, G. & Minkov, M. 2010, *Cultures and organizations: software of the mind*, McGraw-Hill, New York.

Holmstrom, H., Conchúir, E.Ó., Agerfalk, P.J. & Fitzgerald, B. 2006a, 'Global software development challenges: a case study on temporal, geographical and socio-cultural distance', *In Proceedings of the International Conference on Global Software Engineering (ICGSE '06)*, IEEE, pp. 3-11.

Holmström, H., Fitzgerald, B., Ågerfalk, P.J. & Conchúir, E.Ó. 2006b, 'Agile practices reduce distance in global software development', *Information Systems Management*, vol. 23, no. 3, pp. 7-18.

Hossain, E., Babar, M.A. & Verner, J. 2009a, 'Towards a framework for using agile approaches in global software development', in F. Bomarius, M. Oivo, P. Jaring & P. Abrahamsson (eds), *Product-Focused Software Process Improvement*, Springer Berlin Heidelberg, pp. 126-40.

Hossain, E., Babar, M.A. & Paik, H.-Y. 2009b, 'Using Scrum in global software development: a systematic literature review', *Fourth International Conference on Global Software Engineering (ICGSE)*, IEEE, pp. 175-84.

Hossain, E., Bannerman, P.L. & Jeffery, R. 2011, 'Towards an understanding of tailoring scrum in global software development: a multi-case study', *In Proceedings of the 2011 International Conference on Software and Systems Process*, ACM, pp. 110-9.

Hulkko, H. & Abrahamsson, P. 2005, 'A multiple case study on the impact of pair programming on product quality', *In Proceedings of the 27th international conference on Software engineering*, ACM, St. Louis, MO, pp. 495-504.

Hummel, M., Rosenkranz, C. & Holten, R. 2013, 'The role of communication in agile systems development', *Business and Information Systems Engineering*, vol. 5, no. 5, pp. 343-55.

Hummel, M., Rosenkranz, C. & Holten, R. 2015, 'The role of social agile practices for direct and indirect communication in information systems development teams', *Communications of the Association for Information Systems*, vol. 36, no. 1, pp. 273-300.

ISO/IEC 42010 2007, 'Defining architecture', <http://www.iso-architecture.org/ieee-1471/defining-architecture.html>.

ISO/IEC/IEEE 42010 2011, 'Systems and software engineering -- architecture description, viewed 5 December 2013', <http://www.iso-architecture.org/ieee-1471/defining-architecture.html>.

Iivari, J., Hirschheim, R. & Klein, H.K. 1998, 'A paradigmatic analysis contrasting information systems development approaches and methodologies', *Information Systems Research*, vol. 9, no. 2, pp. 164-93.

Jaanu, T., Paasivaara, M. & Lassenius, C. 2012, 'Effects of four distances on communication processes in global software projects', *In proceedings of the ACM-IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM)*, IEEE, pp. 231-4.

Janssen, M. 2011, 'Sociopolitical aspects of interoperability and enterprise architecture in e-government', *Social Science Computer Review*, vol. 000, no. 00, pp. 1-13.

Jarvis, C.B., MacKenzie, S.B. & Podsakoff, P.M. 2003, 'A critical review of construct indicators and measurement model misspecification in marketing and consumer research', *Journal of Consumer Research*, vol. 30, no. 2, pp. 199-218.

Jiang, J. & Klein, G. 2000, 'Software development risks to project effectiveness', *Journal of Systems and Software*, vol. 52, no. 1, pp. 3-10.

Jiménez, M., Piattini, M. & Vizcaíno, A. 2009, 'Challenges and improvements in distributed software development: a systematic review', *Advances in Software Engineering*, vol. 2009, pp. 1-14.

Jonker, J. & Pennink, B. 2010, *The essence of research methodology: a concise guide for master and PhD students in management science*, Springer Science and Business Media.

Kaplan, B. & Duchon, D. 1988, 'Combining qualitative and quantitative methods in information systems research: a case study', *MIS Quarterly*, pp. 571-86.

Karlsen, A. 2008, 'A research model for enterprise modeling in ICT-enabled process change', in J. Stirna & A. Persson (eds), *The Practice of Enterprise Modeling*, Springer Berlin Heidelberg, pp. 217-30.

Keskin, H. 2009, 'Antecedents and consequences of team memory in software development projects', *Information and Management*, vol. 46, no. 7, pp. 388-96.

Khazanchi, D. & Munkvold, B.E. 2003, 'On the rhetoric and relevance of IS research paradigms: a conceptual framework and some propositions', *In Proceedings of the 36th Annual Hawaii International Conference on System Sciences*, IEEE.

Kimmel, A. 1988, *Ethics and values in applied social research*, Sage Publications, Newbury Park, California.

Kirsch, L.J. 1996, 'The management of complex tasks in organizations: controlling the systems development process', *Organization Science*, vol. 7, no. 1, pp. 1-21.

Kitchenham, B. & Charters, S. 2007, 'Guidelines for performing systematic literature reviews in software engineering', *EBSE Technical Report, EBSE-2007-01*.

Klein, M. & Bernstein, A. 2004, 'Toward high-precision service retrieval', *Internet Computing, IEEE*, vol. 8, no. 1, pp. 30-6.

Kline, R.B. 2015, *Principles and practice of structural equation modeling*, 2 edn, Guilford Press, New York.

Ko, D.-G., Kirsch, L.J. & King, W.R. 2005, 'Antecedents of knowledge transfer from consultants to clients in enterprise system implementations', *MIS Quarterly*, pp. 59-85.

Koch, A.S. 2005, *Agile software development: evaluating the methods for your organization*, Artech house Inc., London.

Korhonen, J.J., Lapalme, J., McDavid, D. & Gill, A.Q. 2016, 'Enterprise architecture for the future: towards a reconceptualization of EA', *In the Proceedings of the 18th IEEE Conference on Business Informatics*, Paris.

Korkala, M. & Abrahamsson, P. 2007, 'Communication in distributed agile development: a case study', *In Proceedings of the 33rd Conference on Software Engineering and Advanced Applications (EUROMICRO)*, IEEE, pp. 203-10.

Korkala, M. & Maurer, F. 2014, 'Waste identification as the means for improving communication in globally distributed agile software development', *Journal of Systems and Software*, vol. 95, pp. 122-40.

Korkala, M., Pikkarainen, M. & Conboy, K. 2009, 'Distributed agile development: a case study of customer communication challenges', in P. Abrahamsson, M. Marchesi & F. Maurer (eds), *Agile Processes in Software Engineering and Extreme Programming*, Springer Berlin Heidelberg, pp. 161-7.

Korkala, M., Pikkarainen, M. & Conboy, K. 2010, 'A case study of customer communication in globally distributed software product development', *In Proceedings of the 11th International Conference on Product Focused Software*, ACM, pp. 43-6.

Kornstadt, A. & Sauer, J. 2007, 'Tackling offshore communication challenges with agile architecture-centric development', *In Proceedings of the Working Conference on Software Architecture (WICSA)*, IEEE, pp. 28-31.

Kuusinen, K., Mikkonen, T. & Pakarinen, S. 2012, 'Agile user experience development in a large software organization: good expertise but limited impact', in M. Winckler, P. Forbrig & R. Bernhaupt (eds), *Human-Centered Software Engineering*, Springer Berlin Heidelberg, pp. 94-111.

Lange, M., Mendling, J. & Recker, J. 2015, 'An empirical analysis of the factors and measures of enterprise architecture management success', *European Journal of Information Systems*, pp. 1-21.

Lanubile, F. 2009, 'Collaboration in distributed software development', in A.D. Lucia & F. Ferrucci (eds), *Software Engineering*, Springer Berlin Heidelberg, pp. 174-93.

Lanubile, F., Damian, D. & Oppenheimer, H.L. 2003, 'Global software development: technical, organizational, and social challenges', *ACM SIGSOFT Software Engineering Notes*, vol. 28, no. 6, pp. 1-4.

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

Lawshe, C.H. 1975, 'A quantitative approach to content validity1', *Personnel Psychology*, vol. 28, no. 4, pp. 563-75.

Layman, L., Williams, L., Damian, D. & Bures, H. 2006, 'Essential communication practices for extreme programming in a global software development team', *Information and Software Technology*, vol. 48, no. 9, pp. 781-94.

Lee, G., DeLone, W. & Espinosa, J.A. 2006, 'Ambidextrous coping strategies in globally distributed software development projects', *Communications of the ACM*, vol. 49, no. 10, pp. 35-40.

Lee, J.C., Judge, T.K. & McCrickard, D.S. 2011, 'Evaluating eXtreme scenario-based design in a distributed agile team', *In the Proceedings of the Annual Conference Extended Abstracts on Human Factors in Computing Systems*, ACM, pp. 863-77.

Lee, G. & Xia, W. 2010, 'Toward agile: an integrated analysis of quantitative and qualitative field data', *MIS Quarterly*, vol. 34, no. 1, pp. 87-114.

Lee, S. & Yong, H.-S. 2010, 'Distributed agile: project management in a global environment', *Empirical Software Engineering*, vol. 15, no. 2, pp. 204-17.

Leedy, P.D. & Ormrod, J.E. 2010, *Practical research: planning and design*, 9 edn, Pearson Education, Inc.

Leffingwell, D. 2007, *Scaling software agility: best practices for large enterprises*, 2 edn, Pearson Education.

Lewis, B.R., Templeton, G.F. & Byrd, T.A. 2005, 'A methodology for construct development in MIS research', *European Journal of Information Systems*, vol. 14, no. 4, pp. 388-400.

Leximancer Company 2016, 'Text in insight out', viewed 5 May 2016, <http://info.leximancer.com/>.

Lindstrom, L. & Jeffries, R. 2004, 'Extreme programming and agile software development methodologies', *Information Systems Management*, vol. 21, no. 3, pp. 41-52.

Lissitz, R.W. & Green, S.B. 1975, 'Effect of the number of scale points on reliability: a Monte Carlo approach', *Journal of Applied Psychology*, vol. 60, no. 1, p. 10.

Little, R.J. 1988, 'A test of missing completely at random for multivariate data with missing values', *Journal of the American Statistical Association*, vol. 83, no. 404, pp. 1198-202.

Little, R.J. & Rubin, D.B. 1987, *Statistical analysis with missing data*, John Wiley & Sons, New York.

Lowry, P.B. & Gaskin, J. 2014, 'Partial least squares (PLS) structural equation modeling (SEM) for building and testing behavioral causal theory: when to choose it and how to use it', *IEEE Transactions on Professional Communication*, vol. 57, no. 2, pp. 123-46.

MacKenzie, S.B., Podsakoff, P.M. & Podsakoff, N.P. 2011, 'Construct measurement and validation procedures in MIS and behavioral research: integrating new and existing techniques', *MIS Quarterly*, vol. 35, no. 2, pp. 293-334.

Madison, J. 2010, 'Agile architecture interactions', *IEEE Software*, vol. 27, no. 2, pp. 41-8.

Madni, A.M. 2008, 'AgileTecting™: a principled approach to introducing agility in systems engineering and product development enterprises', *Journal of Integrated Design and Process Science*, vol. 12, no. 4, pp. 49-55.

Mahaney, R.C. & Lederer, A.L. 2006, 'The effect of intrinsic and extrinsic rewards for developers on information systems project success', *Project Management Journal*, vol. 37, no. 4, pp. 42-54.

Malhotra, N.K. & Birks, D.F. 2003, *Marketing research: an applied approach*, 2 edn, Pearson Education, New York.

Malhotra, N.K., Kim, S.S. & Patil, A. 2006, 'Common method variance in IS research: a comparison of alternative approaches and a reanalysis of past research', *Management Science*, vol. 52, no. 12, pp. 1865-83.

Malone, T.W. & Crowston, K. 1990, 'What is coordination theory and how can it help design cooperative work systems?', *In Proceedings of the 1990 ACM conference on Computer-supported cooperative work*, ACM, pp. 357-70.

Marcoulides, G.A., Chin, W.W. & Saunders, C. 2009, 'A critical look at partial least squares modeling', *MIS Quarterly*, vol. 33, no. 1, pp. 171-5.

Marcoulides, G.A. & Saunders, C. 2006, 'Editor's comments: PLS: a silver bullet?', *MIS Quarterly*, vol. 30, no. 2, pp. iii-ix.

Marshall, C. & Rossman, G.B. 2014, *Designing qualitative research*, Sage publications, Thousand Oaks, California.

Martin, R.C. 2003, *Agile software development: principles, patterns, and practices*, Pearson Education, Upper Saddle River, NJ.

Martini, A., Pareto, L. & Bosch, J. 2013, 'Communication factors for speed and reuse in large-scale agile software development', *In Proceedings of the 17th International Software Product Line Conference*, ACM, pp. 42-51.

Maruping, L.M. 2010, 'Implementing extreme programming in distributed software project teams: strategies and challenges', in D. Šmite et al. (eds), *Agility Across Time and Space*, Springer Berlin Heidelberg, pp. 11-30.

Maruping, L.M., Venkatesh, V. & Agarwal, R. 2009, 'A control theory perspective on agile methodology use and changing user requirements', *Information Systems Research*, vol. 20, no. 3, pp. 377-99.

McQuail, D. 1987, *Mass communication theory: an introduction*, Sage Publications, Thousand Oaks, California.

Melo, C., Cruzes, D.S., Kon, F. & Conradi, R. 2011, 'Agile team perceptions of productivity factors', *Agile Conference (AGILE)*, IEEE, pp. 57-66.

Miles, M.B. & Huberman, A.M. 1994, *Qualitative data analysis: an expanded sourcebook*, Sage Publications, Beverly Hills, CA.

Miller, G.R. 1976, *Explorations in interpersonal communication*, Sage Publications, Oxford, England.

Mingers, J. 2001, 'Combining IS research methods: towards a pluralist methodology', *Information Systems Research*, vol. 12, no. 3, pp. 240-59.

Mingers, J. 2003, 'A classification of the philosophical assumptions of management science methods', *Journal of the Operational Research Society*, vol. 54, no. 6, pp. 559-70.

Mishra, D., Mishra, A. & Ostrovska, S. 2012, 'Impact of physical ambiance on communication, collaboration and coordination in agile software development: an empirical evaluation', *Information and Software Technology*, vol. 54, no. 10, pp. 1067-78.

Misra, S.C., Kumar, V. & Kumar, U. 2009, 'Identifying some important success factors in adopting agile software development practices', *Journal of Systems and Software*, vol. 82, no. 11, pp. 1869-90.

MIT 2006, 'Business agility and IT portfolios', MIT Sloan School of Management, Center for Information Systems Research, viewed 28 May 2016, <http://cisr.mit.edu/>.

Modi, S., Abbott, P. & Counsell, S. 2013, 'Negotiating common ground in distributed agile development: a case study perspective', *In Proceedings of the 8th International Conference on Global Software Engineering (ICGSE)*, IEEE, pp. 80-9.

Moe, N.B., Šmite, D., Hanssen, G.K. & Barney, H. 2014, 'From offshore outsourcing to insourcing and partnerships: four failed outsourcing attempts', *Empirical Software Engineering*, vol. 19, no. 5, pp. 1225-58.

Monk, A. 2003, 'Common ground in electronically mediated communication: Clark's theory of language use', in J.M. Carroll (ed.), *HCI models, theories, and frameworks: toward a multidisciplinary science*, Morgan Kaufmann Publisher, pp. 265-89.

Morgeson, F.P. & Humphrey, S.E. 2006, 'The work design questionnaire (WDQ): developing and validating a comprehensive measure for assessing job design and the nature of work', *Journal of Applied Psychology*, vol. 91, no. 6, p. 1321.

Mthupha, B. 2012, 'A framework for the development and measurement of agile enterprise architecture', Master thesis, Rhodes University.

Myers, M.D. 1997, 'Qualitative research in information systems', *MIS Quarterly*, vol. 21, pp. 241-2.

Myers, M.D. & Avison, D. 2002, *Qualitative research in information systems: a reader*, 1 edn, Sage Publications, London.

Nair, S. 2008, 'Agility requirements: the groundwork of agile architecture – Part I', viewed 22 May 2016, <https://earchpal.wordpress.com/2008/07/06/agility-requirements-the-groundwork-of-agile-architecture-%E2%80%93-part-i/>.

Najafi, E. & Baraani, A. 2010, 'KASRA framework: a service oriented enterprise architecture framework (SOEAF)', *In Proceedings of the 6th World Congress on Services*, IEEE, pp. 172-3.

Nerur, S. & Balijepally, V. 2007, 'Theoretical reflections on agile development methodologies', *Communications of the ACM*, vol. 50, no. 3, pp. 79-83.

Nerur, S., Mahapatra, R. & Mangalaraj, G. 2005, 'Challenges of migrating to agile methodologies', *Communications of the ACM*, vol. 48, no. 5, pp. 72-8.

Neuman, W.L. 2003, *Social research methods: quantitative and qualitative approaches*, 5 edn, Allyn and Bacon, Boston, Massachisetts.

Niazi, M., Mahmood, S., Alshayeb, M., Baqais, B., Ahmed, A. & Gill, A.Q. 2013, 'Motivators of Adopting Social Computing in Global Software Development: Initial Results', *World Congress on Engineering and Computer Science*, vol. 1, London, pp. 1-5.

Niemi, E. & Pekkola, S. 2015, 'Using enterprise architecture artefacts in an organisation', *Enterprise Information Systems*, pp. 1-26.

Nunnally, J.C. 1978, *Psychometric theory*, McGraw-Hill, New York.

O'Donoghue, T. & Punch, K. (eds.), 2013, 'Qualitative educational research in action: doing and reflecting', RoutledgeFalmer: Taylor & Francis Group, New York.

O'Leary, M.B., Wilson, J.M. & Metiu, A. 2014, 'Beyond being there: the symbolic role of communication and identification in perceptions of proximity to geographically dispersed colleagues', *MIS Quarterly*, vol. 38, no. 4, pp. 1219-43.

Olson, G.M. & Olson, J.S. 2000, 'Distance matters', *Human-Computer Interaction*, vol. 15, no. 2, pp. 139-78.

Oppenheim, A.N. 2000, *Questionnaire design, interviewing and attitude measurement*, Bloomsbury Publishing, London.

Op't Land, M., Proper, E., Waage, M., Cloo, J. & Steghuis, C. 2008, *Enterprise architecture: creating value by informed governance*, Springer Science and Business Media.

Orlikowski, W.J. & Baroudi, J.J. 1991, 'Studying information technology in organizations: research approaches and assumptions', *Information Systems Research*, vol. 2, no. 1, pp. 1-28.

Paasivaara, M., Durasiewicz, S. & Lassenius, C. 2008, 'Distributed agile development: using Scrum in a large project', *In Proceedings of the International Conference on Global Software Engineering (ICGSE)*, IEEE, pp. 87-95.

Paasivaara, M., Durasiewicz, S. & Lassenius, C. 2009, 'Using Scrum in distributed agile development: a multiple case study', *In Proceedings of the Fourth International Conference on Global Software Engineering (ICGSE)*, IEEE, pp. 195-204.

Paasivaara, M. & Lassenius, C. 2003, 'Collaboration practices in global inter-organizational software development projects', *Software Process: Improvement and Practice*, vol. 8, no. 4, pp. 183-99.

Paasivaara, M. & Lassenius, C. 2010, 'Using Scrum practices in GSD projects', in D. Šmite et al. (eds), *Agility Across Time and Space*, Springer Berlin Heidelberg, pp. 259-78.

Paasivaara, M. & Lassenius, C. 2014, 'Communities of practice in a large distributed agile software development organization–case Ericsson', *Information and Software Technology*, vol. 56, no. 12, pp. 1556-77.

Paasivaara, M., Lassenius, C., Damian, D., Raty, P. & Schroter, A. 2013, 'Teaching students global software engineering skills using distributed Scrum', *In Proceedings of the 35th International Conference on Software Engineering (ICSE)*, IEEE, pp. 1128-37.

Pallant, J. 2001, *SPSS Survival Manual: A Step by Step Guide to Data Analysis Using SPSS for Windows (Versions 10 and 11): SPSS Student Version 11.0 for Windows*, Berkshire: Open University Press.

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

Pare, G. 2004, 'Investigating information systems with positivist case research', *The Communications of the Association for Information Systems*, vol. 13, no. 1, pp. 233-64.

Peng, D.X. & Lai, F. 2012, 'Using partial least squares in operations management research: a practical guideline and summary of past research', *Journal of Operations Management*, vol. 30, no. 6, pp. 467-80.

Petter, S., Straub, D. & Rai, A. 2007, 'Specifying formative constructs in information systems research', *MIS Quarterly*, vol. 31, no. 4, pp. 623-56.

Piccoli, G., Powell, A. & Ives, B. 2004, 'Virtual teams: team control structure, work processes, and team effectiveness', *Information Technology and People*, vol. 17, no. 4, pp. 359-79.

Pikkarainen, M., Haikara, J., Salo, O., Abrahamsson, P. & Still, J. 2008, 'The impact of agile practices on communication in software development', *Empirical Software Engineering*, vol. 13, no. 3, pp. 303-37.

Pinsonneault, A. & Kraemer, K.L. 1993, 'Survey research methodology in management information systems: an assessment', *Journal of Management Information Systems*, pp. 75-105.

Plonka, L., Segal, J., Sharp, H. & van der Linden, J. 2011, 'Collaboration in pair programming: driving and switching', in A. Sillitti, O. Hazzan, E. Bache & X. Albaladejo (eds), *Agile Processes in Software Engineering and Extreme Programming*, Springer Berlin Heidelberg, pp. 43-59.

Podsakoff, P.M., MacKenzie, S.B., Lee, J.-Y. & Podsakoff, N.P. 2003, 'Common method biases in behavioral research: a critical review of the literature and recommended remedies', *Journal of Applied Psychology*, vol. 88, no. 5, pp. 879-903.

Podsakoff, P.M. & Organ, D.W. 1986, 'Self-reports in organizational research: problems and prospects', *Journal of Management*, vol. 12, no. 4, pp. 531-44.

Polzer, J.T., Crisp, C.B., Jarvenpaa, S.L. & Kim, J.W. 2006, 'Extending the faultline model to geographically dispersed teams: how colocated subgroups can impair group functioning', *Academy of Management Journal*, vol. 49, no. 4, pp. 679-92.

Poppendieck, M. & Poppendieck, T. 2013, *The lean mindset*, Viewed 19 December 2013, < http://www.poppendieck.com/>.

Preacher, K.J. & Hayes, A.F. 2004, 'SPSS and SAS procedures for estimating indirect effects in simple mediation models', *Behavior Research Methods, Instruments, and Computers*, vol. 36, no. 4, pp. 717-31.

Preacher, K.J. & Hayes, A.F. 2008, 'Asymptotic and resampling strategies for assessing and comparing indirect effects in multiple mediator models', *Behavior Research Methods*, vol. 40, no. 3, pp. 879-91.

Prikladnicki, R., Marczak, S., Carmel, E. & Ebert, C. 2012, 'Technologies to support collaboration across time zones', *IEEE Software*, no. 3, pp. 10-3.

Proper, E. & Greefhorst, D. 2011, 'Principles in an enterprise architecture context', *Journal of Enterprise Architecture*, vol. 7, no. 1, pp. 8-16.

Puranam, P., Singh, H. & Chaudhuri, S. 2009, 'Integrating acquired capabilities: when structural integration is (un) necessary', *Organization Science*, vol. 20, no. 2, pp. 313-28.

Putney, L.G., Green, J.L., Dixon, C.N. & Kelly, G.J. 1999, 'Evolution of qualitative research methodology: looking beyond defense to possibilities', *Reading Research Quarterly*, vol. 34, no. 3, pp. 368-77.

Qumer, A. & Henderson-Sellers, B. 2006a, 'Measuring agility and adoptability of agile methods: a 4 dimensional analytical tool', *In the Proceedings of the IADIS International Conferences*, pp. 503-7.

Qumer, A. & Henderson-Sellers, B. 2006b, 'Crystallization of agility back to basics', *In Proceedings of the First International Conference on Software and Data Technologies (ICSOFT), INSTICC Press*, vol. 2, pp. 121-6.

Qumer, A. & Henderson-Sellers, B. 2006c, 'Comparative evaluation of XP and Scrum using the 4D Analytical Tool (4-DAT)', *In Proceedings of the European and Mediterranean Conference on Information Systems (EMCIS)*, Costa Blanca, Alicante, Spain, pp. 1-8.

Qumer, A. & Henderson-Sellers, B. 2008, 'An evaluation of the degree of agility in six agile methods and its applicability for method engineering', *Information and Software Technology*, vol. 50, no. 4, pp. 280-95.

Ramasubbu, N., Cataldo, M., Balan, R.K. & Herbsleb, J.D. 2011, 'Configuring global software teams: a multi-company analysis of project productivity, quality, and profits', *In Proceedings of the 33rd International Conference on Software Engineering*, ACM, pp. 261-70.

Ramesh, B., Cao, L. & Baskerville, R. 2010, 'Agile requirements engineering practices and challenges: an empirical study', *Information Systems Journal*, vol. 20, no. 5, pp. 449-80.

Ramesh, B., Cao, L., Mohan, K. & Xu, P. 2006, 'Can distributed software development be agile?', *Communications of the ACM*, vol. 49, no. 10, pp. 41-6.

Ramesh, B., Mohan, K. & Cao, L. 2012, 'Ambidexterity in agile distributed development: an empirical investigation', *Information Systems Research*, vol. 23, no. 2, pp. 323-39.

Ringle, C.M., Wende, S. & Becker, J.-M. 2015, 'SmartPLS 3. SmartPLS GmbH: Boenningstedt', <http://www.smartpls.com>.

Rivera, M.T., Soderstrom, S.B. & Uzzi, B. 2010, 'Dynamics of dyads in social networks: assortative, relational, and proximity mechanisms', *Annual Review of Sociology*, vol. 36, pp. 91-115.

Rogers, E.M. 1986, *Communication technology: the new media in society*, The Free Press, New York.

Ross, J.W., Weill, P. & Robertson, D. 2006, *Enterprise architecture as strategy: creating a foundation for business execution*, 1 edn, Harvard Business Press.

Rowe, F. 2014, 'What literature review is not: diversity, boundaries and recommendations', *European Journal of Information Systems*, vol. 23, no. 3, pp. 241-55.

Sale, J.E., Lohfeld, L.H. & Brazil, K. 2002, 'Revisiting the quantitative-qualitative debate: implications for mixed-methods research', *Quality and Quantity*, vol. 36, no. 1, pp. 43-53.

Sambamurthy, V. & Kirsch, L.J. 2000, 'An integrative framework of the information systems development process', *Decision Sciences*, vol. 31, no. 2, pp. 391-411.

Sarker, S. & Sarker, S. 2009, 'Exploring agility in distributed information systems development teams: an interpretive study in an offshoring context', *Information Systems Research*, vol. 20, no. 3, pp. 440-61.

Sauer, J. 2006, 'Agile practices in offshore outsourcing–an analysis of published experiences', *In Proceedings of the 29th Information Systems Research Seminar in Scandinavia, IRIS*, vol. 29, pp. 12-5.

Sauer, J. 2010, 'Architecture-centric development in globally distributed projects', in D. Šmite et al. (eds), *Agility Across Time and Space*, Springer Berlin Heidelberg, pp. 321-9.

Sawyer, S., Guinan, P.J. & Cooprider, J. 2010, 'Social interactions of information systems development teams: a performance perspective', *Information Systems Journal*, vol. 20, no. 1, pp. 81-107.

Schafer, J.L. 1997, *Analysis of incomplete multivariate data*, Chapman & Hall, New York.

Schulmeisters, K. 2014, 'Enterprise architecture and agile development, contradiction or synergy?', Orbus Software, White Paper, viewed 25 December 2014, <www.orbussoftware.com/community>.

Schümmer, T. & Lukosch, S. 2008, 'Supporting the social practices of distributed pair programming', in R.O. Briggs, P. Antunes, G.-J.d. Vreede & A.S. Read (eds), *Groupware: Design, Implementation, and Use*, Springer Berlin Heidelberg, pp. 83-98.

Schwaber, K. 2004, *Agile project management with Scrum*, Microsoft Press.

Schwaber, K. & Beedle, M. 2002, *Agile software development with Scrum*, Prentice-Hall, Upper Saddle River.

Scotland, K. 2013, 'Aspects of Kanban', Viewed 19 December 2013, <http://availagility.co.uk>.

Scott, N. & Smith, A.E. 2005, 'Use of automated content analysis techniques for event image assessment', *Tourism Recreation Research*, vol. 30, no. 2, pp. 87-91.

Sekaran, U. 2006, *Research methods for business: a skill building approach*, John Wiley & Sons, New York.

Selic, B. 2009, 'Agile documentation, anyone?', *IEEE Software*, vol. 26, no. 6, pp. 11-2.

Shanks, G. 2002, 'Guidelines for conducting positivist case study research in information systems', *Australasian Journal of Information Systems*, vol. 10, no. 1.

Sharp, J.H. 2008, 'Globally distributed agile teams: an exploratory study of the dimensions contributing to successful team configuration', PhD thesis, University of North Texas.

Sharp, H., Giuffrida, R. & Melnik, G. 2012, 'Information flow within a dispersed agile team: a distributed cognition perspective', in C. Wohlin (ed.), *Agile Processes in Software Engineering and Extreme Programming*, Springer Berlin Heidelberg, pp. 62-76.

Sharp, H. & Robinson, H. 2007, 'Collaboration and co-ordination in mature eXtreme programming teams', *International Journal of Human-Computer Studies*, vol. 66, no. 7, pp. 506-18.

Shrivastava, S.V. & Rathod, U. 2015, 'Categorization of risk factors for distributed agile projects', *Information and Software Technology*, vol. 58, pp. 373-87.

Siau, K., Long, Y. & Ling, M. 2010, 'Toward a unified model of information systems development success', *Journal of Database Management (JDM)*, vol. 21, no. 1, pp. 80-101.

Sindhgatta, R., Sengupta, B. & Datta, S. 2011, 'Coping with distance: an empirical study of communication on the jazz platform', *In Proceedings of the International Conference*

Companion on Object Oriented Programming Systems Languages and Applications Companion, ACM, pp. 155-62.

Sivo, S.A., Saunders, C., Chang, Q. & Jiang, J.J. 2006, 'How low should you go? Low response rates and the validity of inference in IS questionnaire research', *Journal of the Association for Information Systems*, vol. 7, no. 6, pp. 351-413.

Smite, D. & Dingsøyr, T. 2012, 'Fostering Cross-site Coordination through Awareness: an investigation of state-of-the-practice through a focus group study', *In Proceedings of the 38th EUROMICRO Conference on Software Engineering and Advanced Applications (SEAA)*, IEEE, pp. 337-44.

Smith, K.G., Smith, K.A., Olian, J.D., Sims Jr, H.P., O'Bannon, D.P. & Scully, J.A. 1994, 'Top management team demography and process: the role of social integration and communication', *Administrative Science Quarterly*, pp. 412-38.

Spector, P.E. 2006, 'Method variance in organizational research truth or urban legend?', *Organizational Research Methods*, vol. 9, no. 2, pp. 221-32.

Stapleton, J. 1997, *DSDM: the method in practice*, Addison-Wesley Longman Publishing Co., Boston, MA.

Stettina, C.J., Heijstek, W. & Fægri, T.E. 2012, 'Documentation work in agile teams: the role of documentation formalism in achieving a sustainable practice', *Agile Conference (AGILE)*, IEEE, pp. 31-40.

Stone, M. 1974, 'Cross-validatory choice and assessment of statistical predictions', *Journal of the Royal Statistical Society. Series B (Methodological)*, pp. 111-47.

Storey, M.-A.D., Čubranić, D. & German, D.M. 2005, 'On the use of visualization to support awareness of human activities in software development: a survey and a framework', *In Proceedings of the 2005 ACM symposium on Software visualization*, ACM, pp. 193-202.

Straub, D.W. 1989, 'Validating instruments in MIS research', *MIS Quarterly*, pp. 147-69.

Straub, D., Boudreau, M.-C. & Gefen, D. 2004, 'Validation guidelines for IS positivist research', *The Communications of the Association for Information Systems*, vol. 13, no. 1, pp. 379-427.

Straub, D.W., Gefen, D. & Boudreau, M.-C. 2005, 'Quantitative research', in D. Avison & J. Pries-Heje (eds), *Research in information systems: a handbook for research supervisors and their students*, vol. 1, Elsevier Butterworth-Heinemann, Burlington, MA, pp. 221-38.

Sutherland, J., Schoonheim, G. & Rijk, M. 2009, 'Fully distributed scrum: replicating local productivity and quality with offshore teams', *In Proceedings of the 42nd Annual Hawaii International Conference on System Sciences (HICSS)* IEEE, pp. 1-8.

Sutherland, J. & Schwaber, K. 2011, 'The Scrum papers: nut, bolts, and origins of an agile framework ', viewed 27 January 2014, <http://jeffsutherland.com/ScrumPapers.pdf>.

Sutherland, J., Viktorov, A., Blount, J. & Puntikov, N. 2007, 'Distributed scrum: agile project management with outsourced development teams', *In Proceedings of the 40th Annual Hawaii International Conference on System Sciences (HICSS)*, IEEE, pp. 274a-84a.

Svensson, H. & Host, M. 2005, 'Views from an organization on how agile development affects its collaboration with a software development team', in F. Bomarius & S. Komi-Sirviö (eds), *Product focused software process improvement*, Springer Berlin Heidelberg, pp. 487-501.

Tabachnick, B.G. & Fidell, L. 2007, *Using multivariate statistics*, Pearson Education, Boston.

Tamm, T., Seddon, P.B., Shanks, G. & Reynolds, P. 2011, 'How does enterprise architecture add value to organisations', *Communications of the Association for Information Systems*, vol. 28, no. 1, pp. 141-68.

Te'eni, D. 2001, 'Review: a cognitive-affective model of organizational communication for designing IT', *MIS Quarterly*, vol. 25, no. 2, pp. 251-312.

The Open Group 2013, 'TOGAF version 9.1 - the open group architecture framework', viewed 5 December 2013, <http://pubs.opengroup.org/architecture/togaf9-doc/arch/>.

The Standish Group 2015, 'CHAOS report 2015', viewed 8 October 2016, <https://www.standishgroup.com/store/services/chaos-report-2015-blue-pm2go-membership.html>.

Thomas, D.M. & Bostrom, R.P. 2010, 'Team leader strategies for enabling collaboration technology adaptation: team technology knowledge to improve globally distributed systems development work', *European Journal of Information Systems*, vol. 19, no. 2, pp. 223-37.

Välimäki, A. & Kääriäinen, J. 2008, 'Patterns for distributed Scrum — a case study', in I.K. Mertins, R. Ruggaber, K. Popplewell & X. Xu (eds), *Enterprise interoperability III*, Springer, London, pp. 85-97.

Vallon, R., Strobl, S., Bernhart, M. & Grechenig, T. 2013, 'Inter-organizational co-development with Scrum: experiences and lessons learned from a distributed corporate development environment', in H. Baumeister & B. Weber (eds), *Agile Processes in Software Engineering and Extreme Programming*, Springer Berlin Heidelberg, pp. 150-64.

Vanhanen, J. & Lassenius, C. 2005, 'Effects of pair programming at the development team level: an experiment', *In Proceedings of the International Symposium on Empirical Software Engineering*, IEEE, Noosa, Australia, pp. 336-45.

Van Waardenburg, G. & Van Vliet, H. 2013, 'When agile meets the enterprise', *Information and Software Technology*, vol. 55, no. 12, pp. 2154-71.

Venkatesh, V., Bala, H., Venkatraman, S. & Bates, J. 2007, 'Enterprise architecture maturity: the story of the veterans health administration', *MIS Quarterly Executive*, vol. 6, no. 2, pp. 79-90.

Venkatesh, V., Brown, S.A. & Bala, H. 2013, 'Bridging the qualitative-quantitative divide: guidelines for conducting mixed methods research in information systems', *MIS Quarterly*, vol. 37, no. 1, pp. 21-54.

Version One 2014, *9th annual state of agile survey*, <http://www.versionone.com>.

Vickoff, J.-P. 2014, 'PUMA agile enterprise architecture', Agile aliance, viewed 28 May 2016, <http://www.entreprise-agile.com/en/Puma-en.htm>.

Vidgen, R. & Wang, X. 2009, 'Coevolving systems and the organization of agile software development', *Information Systems Research*, vol. 20, no. 3, pp. 355-76.

Vlaar, P.W., van Fenema, P.C. & Tiwari, V. 2008, 'Cocreating understanding and value in distributed work: how members of onsite and offshore vendor teams give, make, demand, and break sense', *MIS Quarterly*, vol. 32, no. 2, pp. 227-55.

Vinzi, V.E., Trinchera, L. & Amato, S. 2010, 'PLS path modeling: from foundations to recent developments and open issues for model assessment and improvement', in V.E. Vinzi, W.W. Chin, J. Henseler & H. Wang (eds), *Handbook of partial least squares*, Springer Berlin Heidelberg, pp. 47-82.

Wallace, L., Keil, M. & Rai, A. 2004, 'Understanding software project risk: a cluster analysis', *Information and Management*, vol. 42, no. 1, pp. 115-25.

Walsham, G. 1995, 'The emergence of interpretivism in IS research', *Information Systems Research*, vol. 6, no. 4, pp. 376-94.

Weber, R. 2003, 'Editor's comment: theoretically speaking', *MIS Quarterly*, vol. 27, no. 3, pp. iii-xii.

Weber, R.P. 1990, *Basic content analysis*, 2 edn, Sage Publications.

Wegmann, A., Kotsalainen, A., Matthey, L., Regev, G. & Giannattasio, A. 2008, 'Augmenting the Zachman enterprise architecture framework with a systemic conceptualization', *In Proceedings of the 12th International Conference on Enterprise Distributed Object Computing (CEDOC)*, IEEE, pp. 3-13.

Wilson, N. 2013, 'Agile and Lean – What do they really mean?', Gartner, viewed 28 May 2016, <http://blogs.gartner.com/nathan-wilson/agile-and-lean-what-do-they-really-mean/>.

Wilson, J., Crisp, C.B. & Mortensen, M. 2013, 'Extending construal-level theory to distributed groups: understanding the effects of virtuality', *Organization Science*, vol. 24, no. 2, pp. 629-44.

Working Group 2007, 'Department of Defense Architecture Framework, version 1.5', *Department of Defense, USA*.

Wynn Jr, D. & Williams, C.K. 2012, 'Principles for conducting critical realist case study research in information systems', *MIS Quarterly*, vol. 36, no. 3, pp. 787-810.

Xprogramming 2014, 'What is extreme programming', viewed 27 January 2014, <http://xprogramming.com/what-is-extreme-programming/>.

Yadav, V., Adya, M., Sridhar, V. & Nath, D. 2009, 'Flexible global software development (GSD): antecedents of success in requirements analysis', *Journal of Global Information Management (JGIM)*, vol. 17, no. 1, pp. 1-31.

Yadav, V., Adya, M., Nath, D. & Sridhar, V. 2007, 'Investigating an 'agile-rigid' approach in globally distributed requirements analysis', *In the Proceedings of PACIS 2007, paper 12*.

Yin, R.K. 2009, *Case study research: design and methods*, Sage Publication, Thousands Oaks.

Zachman, J.A. 1987, 'A framework for information systems architecture', *IBM Systems Journal*, vol. 26, no. 3, pp. 276-92.

Zachman, J.A. 2008, 'John Zachman's concise definition of the Zachman framework', Zachman International, Inc., viewed 7 December 2013, <http://zachman.com>.

Zhao, X., Lynch, J.G. & Chen, Q. 2010, 'Reconsidering Baron and Kenny: myths and truths about mediation analysis', *Journal of Consumer Research*, vol. 37, no. 2, pp. 197-206.

Zieris, F. & Salinger, S. 2013, 'Doing Scrum rather than being agile: a case study on actual nearshoring practices', *In Proceedings of the 8th International Conference on Global Software Engineering (ICGSE)*, IEEE, pp. 144-53.

Zikmund, W., Babin, B., Carr, J. & Griffin, M. 2012, *Business research methods*, 9 edn, CENGAGE Learning.

Appendices

Appendix 3.1: GDAD Challenges Systematic Review References

[S1] Agerfalk, P.J., Fitzgerald, B., Holmstrom, H., Lings, B., Lundell, B. & Conchúir, E.O. 2005, 'A framework for considering opportunities and threats in distributed software development', International Workshop on Distributed Software Development, Citeseer, pp. 47-61.

[S2] Agerfalk, P., Fitzgerald, B. & Slaughter, S. 2009, 'Flexible and distributed information systems development: state of the art and research challenges', Information systems research, vol. 20, no. 3, pp. 317-28.

[S3] Ali Babar, M., Ihme, T. & Pikkarainen, M. 2009, 'An industrial case of exploiting product line architectures in agile software development', In the Proceedings of the 13th International Software Product Line Conference, Carnegie Mellon University, pp. 171-9.

[S4] Avritzer, A., Paulish, D., Cai, Y. & Sethi, K. 2010, 'Coordination implications of software architecture in a global software development project', Journal of Systems and Software, vol. 83, no. 10, pp. 1881-95.

[S5] Balijepally, V., Mahapatra, R., Nerur, S. & Price, K.H. 2009, 'Are two heads better than one for software development? The productivity paradox of pair programming', MIS Quarterly, pp. 91-118.

[S6] Batra, D., Xia, W., VanderMeer, D. & Dutta, K. 2010, 'Balancing agile and structured development approaches to successfully manage large distributed software projects: A case study from the cruise line industry', Communications of the Association for Information Systems, vol. 27, no. 1, pp. 379-94.

[S7] Berczuk, S. 2007, 'Back to basics: the role of agile principles in success with a distributed scrum team', Agile Conference (AGILE), IEEE, pp. 382-8.

[S8] Boden, A., Nett, B. & Wulf, V. 2007, 'Coordination practices in distributed software development of small enterprises', Second International Conference on Global Software Engineering, ICGSE 2007, IEEE, pp. 235-46.

[S9] Conchúir, E.Ó., Ågerfalk, P.J., Olsson, H.H. & Fitzgerald, B. 2009, 'Global software development: where are the benefits?', Communications of the ACM, vol. 52, no. 8, pp. 127-31.

[S10] Conboy, K., Coyle, S., Wang, X. & Pikkarainen, M. 2010, 'People over process: key people challenges in agile development', IEEE Software, vol. 28, no. 4, pp. 48–57.

[S11] da Silva, F.Q., Costa, C., França, A.C.C. & Prikladinicki, R. 2010, 'Challenges and solutions in distributed software development project management: a systematic literature review', 5th International Conference onGlobal Software Engineering (ICGSE), IEEE, pp. 87-96.

[S12] Dybå, T. & Dingsøyr, T. 2008, 'Empirical studies of agile software development: a systematic review', Information and software technology, vol. 50, no. 9, pp. 833-59.

[S13] Estler, H., Nordio, M., Furia, C.A., Meyer, B. & Schneider, J. 2012, 'Agile vs. structured distributed software development: a case study', Seventh International Conference on Global Software Engineering (ICGSE), IEEE, pp. 11-20.

[S14] Helquist, J.H., Deokar, A., Meservy, T. & Kruse, J. 2011, 'Dynamic collaboration: participant-driven agile processes for complex tasks', ACM SIGMIS Database, vol. 42, no. 2, pp. 95-115.

[S15] Herbsleb, J.D. 2007, 'Global software engineering: the future of socio-technical coordination', 2007 Future of Software Engineering, IEEE Computer Society, pp. 188-98.

[S16] Herbsleb, J.D. & Mockus, A. 2003, 'An empirical study of speed and communication in globally distributed software development', IEEE Transactions on Software Engineering, vol. 29, no. 6, pp. 481-94.

[S17] Herbsleb, J.D. & Moitra, D. 2001, 'Global software development', IEEE Software, vol. 18, no. 2, pp. 16-20.

[S18] Holmstrom, H., Conchúir, E.Ó., Agerfalk, P.J. & Fitzgerald, B. 2006a, 'Global software development challenges: a case study on temporal, geographical and socio-cultural distance', International Conference on Global Software Engineering, ICGSE 2006, IEEE, pp. 3-11.

[S19] Hossain, E., Babar, M.A. & Paik, H.-y. 2009b, 'Using Scrum in global software development: a systematic literature review', Global Software Engineering, 2009. ICGSE 2009. Fourth International Conference on, IEEE, pp. 175-84.

[S20] Jiménez, M., Piattini, M. & Vizcaíno, A. 2009, 'Challenges and improvements in distributed software development: A systematic review', Advances in Software Engineering, vol. 2009, p. 3.

[S21] Korkala, M. & Abrahamsson, P. 2007, 'Communication in distributed agile development: a case study', 33rd EUROMICRO Conference on Software Engineering and Advanced Applications, IEEE, pp. 203-10.

[S22] Korkala, M., Pikkarainen, M. & Conboy, K. 2010, 'A case study of customer communication in globally distributed software product development', In the Proceedings of the 11th International Conference on Product Focused Software, ACM, pp. 43-6.

[S23] Korkala, M., Pikkarainen, M. & Conboy, K. 2009, 'Distributed agile development: a case study of customer communication challenges', Agile Processes in Software Engineering and Extreme Programming, Springer, pp. 161-7.

[S24] Lanubile, F. 2009, 'Collaboration in distributed software development', Software Engineering, Springer, pp. 174-93.

[S25] Lanubile, F., Damian, D. & Oppenheimer, H.L. 2003, 'Global software development: technical, organisational, and social challenges', ACM SIGSOFT Software Engineering Notes, vol. 28, no. 6, pp. 1-4.

[S26] Layman, L., Williams, L., Damian, D. & Bures, H. 2006, 'Essential communication practices

for Extreme Programming in a global software development team', Information and Software Technology, vol. 48, no. 9, pp. 781-94.

[S27] Lee, G., DeLone, W. & Espinosa, J.A. 2006, 'Ambidextrous coping strategies in globally distributed software development projects', Communications of the ACM, vol. 49, no. 10, pp. 35-40.

[S28] Lee, J.C., Judge, T.K. & McCrickard, D.S. 2011, 'Evaluating eXtreme scenario-based design in a distributed agile team', In the Proceedings of the Annual Conference Extended Abstracts on Human Factors in Computing Systems, ACM, pp. 863-77.

[S29] Lee, S. & Yong, H.-S. 2010, 'Distributed agile: project management in a global environment', Empirical Software Engineering, vol. 15, no. 2, pp. 204-17.

[S30] Mishra, D., Mishra, A. & Ostrovska, S. 2012, 'Impact of physical ambiance on communication, collaboration and coordination in agile software development: an empirical evaluation', Information and Software Technology, vol. 54, no. 10, pp. 1067-78.

[S31] Niazi, M., Mahmood, S., Alshayeb, M., Baqais, B., Ahmed, A. & Gill, A.Q. 2013, 'Motivators of Adopting Social Computing in Global Software Development: Initial Results', World Congress on Engineering & Computer Science, vol. 1, London, pp. 1-5.

[S32] Paasivaara, M., Durasiewicz, S. & Lassenius, C. 2009, 'Using scrum in distributed agile development: a multiple case study', Fourth International Conference on Global Software Engineering, ICGSE 2009, IEEE, pp. 195-204.

[S33] Paasivaara, M. & Lassenius, C. 2010, 'Using Scrum practices in GSD projects', Agility Across Time and Space, Springer, pp. 259-78.

[S34] Paasivaara, M., Lassenius, C., Damian, D., Raty, P. & Schroter, A. 2013, 'Teaching students global software engineering skills using distributed Scrum', 35th International Conference on Software Engineering (ICSE), IEEE, pp. 1128-37.

[S35] Ramesh, B., Cao, L., Mohan, K. & Xu, P. 2006, 'Can distributed software development be agile?', Communications of the ACM, vol. 49, no. 10, pp. 41-6.

[S36] Sharp, H., Giuffrida, R. & Melnik, G. 2012, 'Information flow within a dispersed agile team: a distributed cognition perspective', Agile Processes in Software Engineering and Extreme Programming, Springer, pp. 62-76.

[S37] Sharp, H. & Robinson, H. 2008, 'Collaboration and co-ordination in mature eXtreme programming teams', International Journal of Human-Computer Studies, vol. 66, no. 7, pp. 506-18.

Appendix 3.2: GDAD Communication Challenges Systematic Review References - Empirical GDAD Studies

Refer to (Alzoubi, Gill & Al-Ani 2016)

[S1] Bannerman, P.L., Hossain, E. & Jeffery, R. 2012, 'Scrum practice mitigation of global

software development coordination challenges: a distinctive advantage?', 45th Hawaii International Conference on System Science (HICSS), IEEE, pp. 5309-18.

[S2] Boden, A., Nett, B. & Wulf, V. 2007, 'Coordination practices in distributed software development of small enterprises', Second International Conference on Global Software Engineering, ICGSE 2007, IEEE, pp. 235-46.

[S3] Bosch, J. & Bosch-Sijtsema, P. 2010, 'Coordination between global agile teams: From process to architecture', Agility Across Time and Space, Springer, pp. 217-33.

[S4] Dorairaj, S., Noble, J. & Malik, P. 2011, 'Effective communication in distributed Agile software development teams', Agile Processes in Software Engineering and Extreme Programming, Springer, pp. 102-16.

[S5] Gill, A.Q. & Bunker, D. 2013, 'Towards the development of a cloud-based communication technologies assessment tool: An analysis of practitioners' perspectives', VINE, vol. 43, no. 1, pp. 57-77.

[S6] Green, R., Mazzuchi, T. & Sarkani, S. 2010, 'Communication and quality in distributed agile development: an empirical case study', Proceeding in World Academy of Science, Engineering and Technology, vol. 61, pp. 322-8.

[S7] Holmström, H., Fitzgerald, B., Ågerfalk, P.J. & Conchúir, E.Ó. 2006b, 'Agile practices reduce distance in global software development', Information Systems Management, vol. 23, no. 3, pp. 7-18.

[S8] Korkala, M. & Abrahamsson, P. 2007, 'Communication in distributed agile development: a case study', 33rd EUROMICRO Conference on Software Engineering and Advanced Applications, IEEE, pp. 203-10.

[S9] Korkala, M. & Maurer, F. 2014, 'Waste identification as the means for improving communication in globally distributed agile software development', Journal of Systems and Software, vol. 95, pp. 122-40.

[S10] Korkala, M., Pikkarainen, M. & Conboy, K. 2010, 'A case study of customer communication in globally distributed software product development', In the Proceedings of the 11th International Conference on Product Focused Software, ACM, pp. 43-6.

[S11] Layman, L., Williams, L., Damian, D. & Bures, H. 2006, 'Essential communication practices for Extreme Programming in a global software development team', Information and Software Technology, vol. 48, no. 9, pp. 781-94.

[S12] Lee, S. & Yong, H.-S. 2010, 'Distributed agile: project management in a global environment', Empirical Software Engineering, vol. 15, no. 2, pp. 204-17.

[S13] Maruping, L.M. 2010, 'Implementing Extreme Programming in distributed software project teams: strategies and challenges', Agility Across Time and Space, Springer, pp. 11-30.

[S14] Paasivaara, M., Lassenius, C., Damian, D., Raty, P. & Schroter, A. 2013, 'Teaching students global software engineering skills using distributed scrum', 35th International Conference on Software Engineering (ICSE), IEEE, pp. 1128-37.

[S15] Paasivaara, M., Durasiewicz, S. & Lassenius, C. 2009, 'Using scrum in distributed agile development: a multiple case study', Fourth International Conference on Global Software Engineering, ICGSE 2009, IEEE, pp. 195-204.

[S16] Sharp, H., Giuffrida, R. & Melnik, G. 2012, 'Information flow within a dispersed agile team: a distributed cognition perspective', Agile Processes in Software Engineering and Extreme Programming, Springer, pp. 62-76.

[S17] Sindhgatta, R., Sengupta, B. & Datta, S. 2011, 'Coping with distance: an empirical study of communication on the jazz platform', In the Proceedings of the International Conference Companion on Object Oriented Programming Systems Languages and Applications Companion, ACM, pp. 155-62.

[S18] Välimäki, A. & Kääriäinen, J. 2008, 'Patterns for distributed Scrum — a case study', Enterprise interoperability III, Springer, pp. 85-97.

[S19] Vallon, R., Strobl, S., Bernhart, M. & Grechenig, T. 2013, Inter-organisational co-development with Scrum: experiences and lessons learned from a distributed corporate development environment, Springer.

[S20] Yadav, V., Adya, M., Nath, D. & Sridhar, V. 2007, 'Investigating an'Agile-Rigid'Approach in Globally Distributed Requirements Analysis', In the Proceedings of PACIS 2007, paper 12.

[S21] Zieris, F. & Salinger, S. 2013, 'Doing Scrum rather than being agile: a case study on actual nearshoring practices', 8th International Conference on Global Software Engineering, ICGSE 2013, IEEE, pp. 144-53.

Appendix 5.1: List of Initial Generated Items

Appendix 5.1(a): Initial Pool of Items

Construct	Literature	Questions/Items
Enterprise Architecture (EA)	Ambler 2011 (1=very effectively, 2= effectively, 3= neutral, 4= ineffectively, 5= very ineffectively, 6= don't know)	1. How well does your organisation's EA group work with agile teams? 2. How agile is your EA team in the way that it works? 3. How does your EA team view agile strategies?
	CIO 2010	4. Which Organisational Model of EA best fits your organisation? (e.g., infrastructure, business…) 5. If you have architect or EA group, where does your EA group report? (e.g., manager, director…) 6. Which choice most closely matches their range of effect? (e.g., group, department…) 7. When do they usually engage with projects? (e.g., at inception, during requirements gathering…)

		8. How would you best describe their method of engagement (e.g., making recommendation, reviewing requirements…) 9. Does the project management office / portfolio management process work closely with EA? (Yes, no, don't have EA groups) 10. Does the project management office / portfolio management process help reinforce EA? (Yes, no, don't have EA groups)
	Franke et al. 2010 (1 very low, 5 very high)	11. My company uses EA models 12. My company uses EA tools 13. My company uses EA processes 14. My company uses EA roles
	Adapted from OCIO 2007 (7 strongly disagree, 1 strongly agree)	15. Enterprise architecture should be shared among IT staff and business management. 16. Enterprise architecture should focus on analysis of the project gaps. 17. Enterprise architecture should be used to assess major project investment. 18. Enterprise architecture process supports the enterprise-wide architecture process of project. 19. Enterprise architecture process actively engages IT projects. 20. Enterprise architecture should be documented
Communication Efficiency	Herbsleb & Mockus 2003 (5 strongly disagree, 1 strongly agree)	1. I lose time trying to figure out who to contact regarding my work 2. People I need to communicate with are difficult to find 3. I often get timely information about changes in current plans
	Misra et al. 2009 (5 strongly disagree, 1 strongly agree)	4. Our organisation encourages rapid communication 5. Our projects have mechanisms that enable personnel to communicate quickly with developers, operations, support, customers, management, and business areas 6. The majority of our team members have strong interpersonal and communication skills 7. Most people in our projects are amicable to each other to such an extent that they communicate with each other with trust and good will
	People Pulse 2012 (5 strongly disagree, 1 strongly agree)	8. I often do not get timely information due to high cost of communication 9. The communication tools which I use provides quick communication 10. I am satisfied with speed of reaction from colleagues from other sites 11. I feel comfortable communicating with colleagues from other sites 12. Meetings are well organised 13. Written directives are well written 14. Communication tools which I use provide a rich information media

	Piccoli et al. 2004 (7 strongly disagree, 1 strongly agree)	15. Most of the information I receive on a daily basis is timely
Communication Effectiveness	Herbsleb & Mockus 2003	1. I often get useful work-related information through casual conversations 2. When work is assigned, everyone is clear about his or her task
	Jiang & Klein 2000	3. Communications between those involved in the project are unpleasant
	Misra et al. 2009 (5 strongly disagree, 1 strongly agree)	4. Our projects have mechanisms that enable personnel to communicate effectively with developers, operations, support, customers, management, and business areas
	Morgeson & Humphrey 2006 (5 strongly disagree, 1 strongly agree)	5. I receive a great deal of information from my manager and co-workers about my job performance. 6. Other people in the organisation, such as managers and co-workers, provide information about the effectiveness (e.g., quality and quantity) of my job performance. 7. I receive feedback on my performance from other people in my organisation (such as my manager or co-workers).
	Piccoli et al. 2004 (7 strongly disagree, 1 strongly agree)	8. Most of the information I receive on a daily basis is accurate 9. Most of the information I receive on a daily basis is reliable 10. Most of the information I receive on a daily basis is useful
	Yadav et al. 2009 (7 strongly disagree, 1 strongly agree)	11. We had frequent online-meetings with MU team members for coordination and planning. 12. We had frequent online-meetings with MU team members for requirements analysis. 13. We had frequent online formal review meetings with MU team members. 14. I participated actively in the online discussions with MU team members.
Software Functionality	Lee & Xia 2010 (7 strongly disagree, 1 strongly agree)	1. The software delivered by the project achieved its functional goals 2. The software delivered by the project met end-user requirements 3. The capabilities of the software fit end-user needs 4. The software met technical requirements
	Mahaney & Lederer 2006 (5 strongly disagree, 1 strongly agree)	5. The project that has been developed works 6. The project is used by its intended users 7. This project has directly benefited the intended users either through increasing efficiency or employee effectiveness 8. Important clients, directly affected by this project,

		will make use of it 9. We are confident that non-technical start-up problems will be minimal, because the project will be readily accepted by its intended users
On-Time Completion	Jiang & Klein 2000	1. In order to develop and implement the system, the scheduled number of people-day is insufficient
	Lee & Xia 2010	2. The project was completed late according to the original schedule (7 strongly disagree, 1 strongly agree) 3. Project start date: _____, planned completion date: _____, actual completion date _____
	Mahaney & Lederer 2006	4. The project came in within its original schedule (5 strongly disagree, 1 strongly agree)
	Misra et al. 2009 (5 strongly disagree, 1 strongly agree)	5. We deliver working software more frequently, from couple of weeks to a couple of months, with a preference to a shorter timescale
	Yadav et al. 2009 (7 strongly disagree, 1 strongly agree)	6. The virtual team project was completed within its original schedule 7. The MU team members insisted on complete and on-time submission of final project deliverables
On-Budget Completion	Jiang & Klein 2000	1. In order to develop and implement the system, the dollar budget provided is insufficient
	Lee & Xia 2010	2. The project was completed over budget according to the original budget (7 strongly disagree, 1 strongly agree) 3. Planned project cost: (in dollar) _____, actual project cost: (in dollar)
	Mahaney & Lederer 2006	4. The project came in within its original budget (5 strongly disagree, 1 strongly agree)
Software Quality	Mahaney & Lederer 2006 (5 strongly disagree, 1 strongly agree)	1. Given the problem for which it was developed, this project seems to do the best job of solving that problem, i.e., it was the best choice among the set of alternatives 2. Use of this project has directly led to improved or more effective decision making or performance for the clients 3. This project will have a positive impact on those who make use of it 4. The results of this project represent a definite improvement over the way the clients used to perform these activities
	Misra et al. 2009 (5 strongly disagree, 1 strongly agree)	5. We give high priority to satisfying customers through early and continuous delivery of valuable software 6. We welcome changing requirements, even late during development

Appendix 5.1(b): Initial Set of Instrument's Items Sent to Experts

Construct	Questions/Items	Action
Enterprise Architecture (EA)	1. How well does your organisation's EA group work with agile teams? 2. How agile is your EA team in the way that it works? 3. How does your EA team view agile strategies? 4. Which choice most closely matches their range of effect? (e.g., group, department…) 5. When do they usually engage with projects? (e.g., at inception, during requirements gathering…) 6. How would you best describe their method of engagement (e.g., making recommendation, reviewing requirements…) 7. Does the project management office / portfolio management process work closely with EA? (Yes, no, don't have EA groups) 8. Does the project management office / portfolio management process help reinforce EA? (Yes, no, don't have EA groups) 9. Enterprise architecture should be shared among IT staff and business management. 10. Enterprise architecture should focus on analysis of the project gaps. 11. Enterprise architecture should be used to assess major project investment. 12. Enterprise architecture process supports the enterprise-wide architecture process of project. 13. Enterprise architecture process actively engages IT projects. 14. Enterprise architecture should be documented.	All these items were deleted and new 7 items were created according to the experts' feedback
Communication Efficiency	1. I lose time trying to figure out who to contact regarding my work 2. People I need to communicate with are difficult to find 3. I often get timely information about changes in current plans 4. Our organisation encourages rapid communication 5. The majority of our team members have strong interpersonal and communication skills 6. I often do not get timely information due to high cost of communication 7. The communication tools which I use provides quick communication 8. I am satisfied with speed of reaction from colleagues from other sites 9. I feel comfortable communicating with	Questions: 1, 2, 3, and 4 were retained Questions: 5, 7, 8, and 9 were deleted Question 6 was reworded. Another 2 questions were added: 1. The people needed to communicate with are reached quickly

	colleagues from other site-deleted	2. The people needed to communicate with are reached easily
Communication Effectiveness	1. I often get useful work-related information through casual conversations 2. When work is assigned, everyone is clear about his or her task 3. I receive a great deal of information from my manager and co-workers about my job performance 4. I receive feedback on my performance from other people in my organisation (such as my manager or co-workers) 5. Most of the communication I do on a daily basis is accurate 6. Most of the communication I do on a daily basis is reliable 7. Most of the communication I do on a daily basis is useful 8. I participated actively in the online discussions with other team members	Questions: 1, 2, and 8 were retained Questions: 3 and 4 were deleted Questions: 5, 6, and 7 were reworded
On-Time Completion	1. The project was completed late according to the original schedule 2. Project start date: _____, planned completion date: _____, actual completion date _____ 3. We deliver working software more frequently, from couple of weeks to a couple of months, with a preference to a shorter timescale 4. The virtual team project was completed within its original schedule	Questions: 1 and 2 were retained as they are Question 3 was deleted Question 4 was reworded
On-Budget Completion	1. In order to develop and implement the system, the dollar budget provided is insufficient 2. The project was completed over budget according to the original budget 3. Planned project cost: (in dollar) _____, actual project cost: (in dollar)	Questions: 2 and 3 were retained as they are Question 1 was deleted Another 1 question was added: The team members complete their tasks according to the original budget
Software Functionality	1. The software delivered by the project achieved its functional goals 2. The software delivered by the project met end-user requirements 3. The capabilities of the software fit end-user needs	Questions: 1, 2, 3, 4, 5, 7, and 8 were retained as they are

	4. The software met technical requirements 5. The project that has been developed works 6. The project is used by its intended users 7. This project has directly benefited the intended users either through increasing efficiency or employee effectiveness 8. Important clients, directly affected by this project, will make use of it	Question 6 was deleted
Software Quality	1. Given the problem for which it was developed, this project seems to do the best job of solving that problem, i.e., it was the best choice among the set of alternatives 2. Use of this project has directly led to improved or more effective decision making or performance for the clients 3. This project will have a positive impact on those who make use of it 4. The results of this project represent a definite improvement over the way the clients used to perform these activities	All the four questions: 1, 2, 3, and 4 were retained as they are

Appendix 5.2: The Full-Scale Survey Questionnaire

Appendix 5.2(a): Full-Scale Consent and Information Form (The Ethics Form)

CONSENT FORM

I _____ (participant's name) agree to participate in the research project: Communication in Distributed Agile Development, being conducted by Yehia Alzoubi of the University of Technology, Sydney for his PhD degree requirements.

I understand that the purpose of this study is to provide insights into how enterprise architecture can enhance the communication in geographically distributed agile development.

I understand that I have been asked to participate in this research because I am an

agile member or an agile architect and my organisation use agile enterprise architecture. My participation in this research will involve my experience about the extent to which the communication between different distributed teams and team members has been effective and efficient. Participation in this study is voluntary, research results will be published in academic and industry outlets but individual information and results specific to any organisation and individual will remain confidential and anonymous. No costs will be incurred by either your organisation or you.

I am aware that I can contact Yehia Alzoubi or his supervisor(s) Bruce Moulton or Asif Gill if I have any concerns about the research. I also understand that I am free to withdraw my participation from this research project at any time I wish, without consequences, and without giving a reason.

I agree that Yehia Alzoubi has answered all my questions fully and clearly.

I agree that the research data gathered from this project may be published in a form that does not identify me in any way.

_____ ____/____/____

Signature (participant)

Yehia Alzoubi

_____ ____/____/____

Signature (researcher or delegate)

NOTE:

This study has been approved by the University of Technology, Sydney Human Research Ethics Committee. If you have any complaints or reservations about any aspect of your participation in this research which you cannot resolve with the researcher, you may contact the Ethics Committee through the Research Ethics Officer (ph: +61 2 9514 9772 Research.Ethics@uts.edu.au) and quote the UTS HREC reference number. Any complaint you make will be treated in confidence and investigated fully and you will be informed of the outcome.

INFORMATION SHEET AND CONSENT FORM FOR ONLINE SURVEYS

Distributed Agile Development Communication (Ethics Approval: 2013000415)

My name is Yehia Alzoubi and I am a student at UTS. (My supervisor is Dr Bruce Moulton)

The purpose of this research /online survey is to find out if the use of enterprise architecture enhances the communication in geographically distributed agile development (communication among dispersed teams and members).

I will ask you to participate in this questionnaire and answer all questions, which need about 10-15 minutes to finish.

You can change your mind at any time and stop completing the survey without consequences.

If you agree to be part of the research and to research data gathered from this survey to be published in a form that does not identify you, please continue with answering the survey questions.

If you have concerns about the research that you think I or my supervisor can help you with, please feel free to contact me (us) on: Yehia Alzoubi, yehia.i.alzoubi@student.uts.edu.au, +61404767612, OR Bruce Moultonl, Bruce.Moulton@uts.edu.au , +61 2 9514 2681

If you would like to talk to someone who is not connected with the research, you may contact the Research Ethics Officer on 02 9514 977 or Research.ethics@uts.edu.au and quote this number **(2013000415)**

Appendix 5.2(b): Full-Scale Questionnaire

UNIVERSITY OF TECHNOLOGY SYDNEY

Geographically Distributed Agile Development Communication Questionnair

Introduction

This survey asks questions about communication between teams or team members who use distributed agile development, the role of enterprise architecture in this communication, and the effect of enterprise architecture use and communication on project performance. Enterprise architecture, communication, and project performance may go by different titles in different organisations. Please refer to the following definitions while completing this survey.

Agile development: Software development that rapidly creates change, proactively or reactively embraces change, and learns from change while contributing to perceived customer value. Scrum and XP are two examples of agile methods.

Geographically distributed agile development (GDAD): Agile development that uses distributed teams or team members in different geographical locations and different time-zones to deliver a certain project.

Enterprise architecture (EA): A blueprint that describes the overall structural, behavioural, social, technological, and facility elements of an enterprise's operating environment that share common goals and principles. Enterprise architecture includes different architecture domains such as Application architecture, Platform architecture, Infrastructure architecture, Business architecture, Solution architecture, and

Information architecture.

Communication: Exchanging information or messages between two parties (i.e. sender and receiver). Good communication should be **efficient** and **effective**.

There are four parts to this survey:

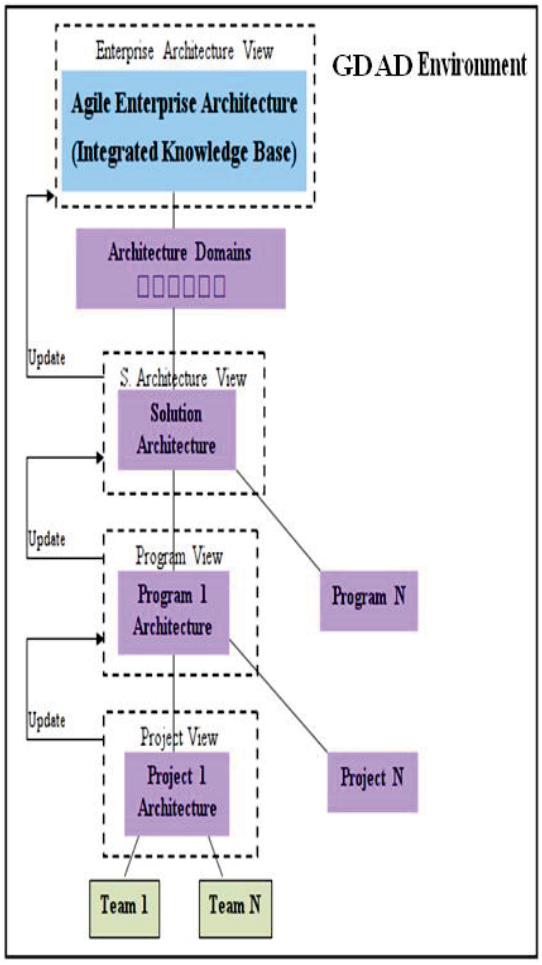
1. The first part focuses on **enterprise architecture (EA)** use between geographically distributed teams or team members.
2. The second part focuses on **communication (efficiency and effectiveness)** between geographically distributed teams or team members.
3. The third part focuses on **software performance** delivered by geographically distributed teams or team members.
4. The final section asks some **demographic** questions about participants.

Study Approach

Using enterprise architecture in a co-located team, whilst still hard, is much easier than in geographically distributed agile development environment. This is because, for a co-located team, there is an opportunity for daily face-to-face communication at any time, which helps in dealing with questions, confusions, difficulties, or doubts instantaneously.

Interaction between teams who use geographically distributed agile development is harder due to many challenges (e.g., different time-zones, cultures, languages, and work practices). This needs asynchronous communication or travelling to meet face-to-face with other teams or team's members.

In geographically distributed agile development environment, agile enterprise architecture may affect the distributed agile development by providing a comprehensive vision through one common object of work that all stakeholders (architects, team members, etc.) in geographically distributed agile development environment use and understand. Agile enterprise architecture thus can be used as an integrated "knowledge base" in geographically distributed agile development environment. Here, we will refer to this approach as enterprise architecture (EA) driven geographically distributed agile development (GDAD) approach, or (**EA driven GDAD approach**).



This diagram explains how "**EA driven GDAD approach**" can be used in GDAD environment.

- As the diagram shows, we have different architectural views according to different architectural (enterprise project management levels).

1. Distributed teams (up to N teams) share the "project architecture view".
2. Different projects (up to N projects) Share the "program architecture view". The same is applied to the "solution architecture view", which can have "N" number of program architectures.
3. Each architecture updates the architecture above. All architectures are then updated and shared using the enterprise architecture integrated knowledge-base. This knowledge-base can be represented in multiple repositories which grant access to all distributed stakeholders. This way ensures that all stakeholders are updated with the latest changes (i.e. project or program changes).

Note:

This survey is to be completed by an agile team member (e.g., project manager, team lead/Scrum Master, developer, operations/support, QA/test) or an agile architect in an organisation that employ agile EA.

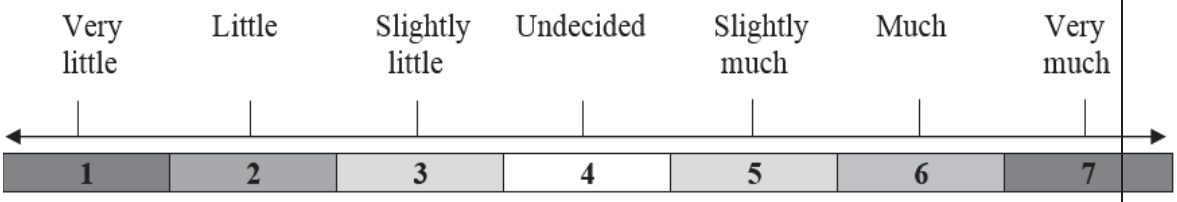
By continuing, we ask gently that:

1. Your answer should be the level of agreement with the situations introduced by each issue, based on your own experience.
2. In case you do not feel comfortable in terms of knowledge to answer precisely any of the questions, please, do not leave unanswered. Feel comfortable in providing approximate answers. For the research, it is most important that you give an approximate answer than no response at all.

Part 1: Agile Enterprise Architecture

Enterprise architecture is a blueprint that describes the overall structural, behavioural, social, technological, and facility elements of an enterprise's operating environment that share common goals and principles.

Listed below are a number of statements about using **enterprise architecture (EA)** in **geographically distributed agile development (GDAD)** projects. Please indicate the extent to which you personally agree or disagree with each of these statements using the following scale (1-7).



By using "EA driven GDAD approach" (as introduced above):

1. Enterprise architecture framework and the framework of GDAD are aligned __
2. Enterprise architecture documentations are regularly updated to align with the projects in GDAD __
3. Enterprise architecture is used to define projects/programs (e.g., business/IT gap analysis) in geographically distributed agile development __
4. Enterprise architecture is used to assess major project investment in GDAD __
5. Solution architecture, as a part of enterprise architecture, guides the projects at program levels and project levels in GDAD __
6. Solution architecture evolves from small iterations, and the changes in solution architecture are reflected in enterprise architecture __
7. Enterprise architecture is used to govern project implementation in GDAD __

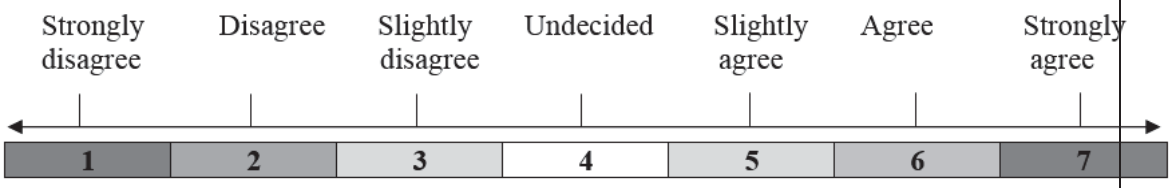
Comments:

_____

_____

Part 2: Communication

Listed below are a number of statements about the communication in geographically distributed agile development (GDAD). Please indicate the extent to which you personally agree or disagree with each of these statements when communication between dispersed teams or team members takes place. Please use the following scale (1-7).



Communication Efficiency

Communication efficiency refers to delivering high quality messages with minimal time, cost, effort, and resources.

By using “EA driven GDAD approach”:

1. Information needed about GDAD project is achieved quickly __
2. Information needed about GDAD project is achieved easily __
3. The stakeholders needed to communicate with in GDAD are reached quickly __
4. The stakeholders needed to communicate with in GDAD are reached easily __
5. The cost of communication (e.g., less travelling to meet face-to-face) in GDAD is decreased __

Communication Effectiveness

Communication effectiveness refers to delivering a message as it was intended with minimal disruption and misunderstanding, even if it takes a long time.

By using “EA driven GDAD approach”:

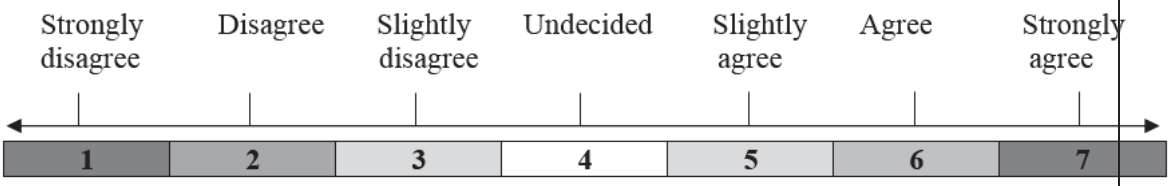
1. All GDAD team members are clear about their tasks __
2. Enough information is provided about customer requirements and project progress to GDAD team members __
3. Detailed information is provided from distributed stakeholders __
4. Accurate information is provided from distributed stakeholders __

Comments:

Part 3: Project Performance

Project performance refers to four dimensions; on-time completion, on-time budget, functionality, and quality of the software developed.

To what extent do you agree with the following statements from your experience of working in distributed agile development? Please use the following scale (1-7).



Functionality (scope)

Functionality refers to the extent to which the delivered project meets its functional goals, user needs, and technical requirements.

By using "EA driven GDAD approach":

1. GDAD project achieves its functional goals __
2. GDAD project meets its technical functional requirements __
3. GDAD project meets customer's functional requirements __

Quality

Quality refers to how good the work is (according to ISO 8402, it is "the totality of characteristics of an entity that bear on its ability to satisfy stated and implied needs").

By using "EA driven GDAD approach":

1. GDAD project solves the given problem __
2. GDAD project improves the way of customers' use to perform their activities __
3. GDAD project achieves customer's satisfaction __

On-time completion

On-time completion refers to the extent to which a software project meets its baseline goals for duration.

By using "EA driven DAD approach":

1. GDAD project is completed on-time according to the original schedule __
2. GDAD tams complete their tasks on-time according to the original schedule __

On-budget completion

On-budget completion refers to the extent to which a software project meets its baseline goals for cost.

By using "EA driven DAD approach":

1. GDAD project is completed on-budget according to the original budget __
2. GDAD tams complete their tasks on-budget according to the original budget __

Comments:

_____

_____

_____

_____

_____

_____

_____

_____

_____

_____

_____

_____

_____

Part 4: Demographics

Following are some questions about you and your organisation. Please check only one answer for each question (you may check a box using "X" or "*").

1. Which best describes your current position?

- ☐ Developer
- ☐ Team lead/Scrum Master
- ☐ Analyst
- ☐ Project manager
- ☐ Other _____
- ☐ Business stakeholder
- ☐ Operations/support
- ☐ Architect
- ☐ QA/test

2. How many years of work experience in IT/systems do you have?

- ☐ None
- ☐ 2 -4 years
- ☐ 10+ years
- ☐ Less than 2 years
- ☐ 5- 10 years

3. How many years of work experience at agile software development do you have?

- ☐ None
- ☐ 2 -4 years
- ☐ 10+ years
- ☐ Less than 2 years
- ☐ 5- 10 years

4. Usually, number of people in your team (counting all included such as IT, Systems, Development, Etc.)

- ☐ 1 - 3
- ☐ 11 - 20
- ☐ 31 - 50
- ☐ 100 +
- ☐ 4 - 10
- ☐ 21 - 30
- ☐ 51 - 100

5. Most of the projects that I have been involved in consist of

- [ ] 1team
- [ ] 2 teams
- [ ] 3 – 5 teams
- [ ] 6 – 10 teams
- [ ] 10 – 15 teams
- [ ] 15 + teams

6. Your organisation's industry is

- [ ] Banking and financial
- [ ] Automotive
- [ ] Telecommunications
- [ ] Health care and medical
- [ ] Other _____

7. How would you describe your organisation's approach when acquiring solutions?

- [ ] Some best-of-breed some single vendor integrated solutions
- [ ] We pick best-of-breed solution
- [ ] We pick single-vendor integrated solutions
- [ ] Other _____

8. Do many of your programming activities include use of open source software (OSS) products?

- [ ] Significant portions are open source software based
- [ ] Few portions are open source softw
- [ ] Some portions are open source software based
- [ ] No portions are open source softwa

Thanks very much for your time and effort

If you have any comments on the survey, please feel free to write them below

Comments:

_____

_____

_____

_____

_____

_____

_____

_____

_____

_____

_____

_____

_____

Note:

Please leave your email if you would like to receive a copy of the final findings

_____

Appendix 5.3: The Interview Protocol

Appendix 5.3(a): Interview Consent Form

CONSENT FORM

I _____ (participant's name) agree to participate in the research project: An Architecture-Driven Communication for Distributed Agile Development, being conducted by Yehia Alzoubi of the University of Technology, Sydney for his PhD degree requirements.

I understand that the purpose of this study is to evaluate the final results and model for communication in distributed agile development. The student got his final results and model after conducting a quantitative study.

I am aware that my participation in this research will involve evaluation of the

communication research model in distributed agile development environment, the research design and research methodology that the student follows, and the final results that the student got after conducting a quantitative analysis.

I understand my role in evaluating the model will involve completing an evaluation form at the end of this document which includes three sections: demographic questions, general questions about the research design, methodology, and model versions, and questions about the final results and model. I understand that the evaluation will be based on a ranking scale and that there will also be comment boxes if I wish to add specific comments.

I am aware that this evaluation will take about 30 - 60 minutes. Also, I know the potential risk associated with this research is minimal due to the nature of the evaluation (which will be conducted by email).

I am aware that I can contact Yehia Alzoubi or his supervisor(s) Bruce Moulton or Asif Gill if I have any concerns about the research. I also understand that I am free to withdraw my participation from this research project at any time I wish, without consequences, and without giving a reason.

I agree that the research data gathered from this project may be published in a form that does not identify me in any way.

_____

____/____/____

Signature (participant)

Yehia Alzoubi

_____

____/____/____

Signature (researcher or delegate)

NOTE:

This study has been approved by the University of Technology, Sydney Human Research Ethics Committee. If you have any complaints or reservations about any aspect of your participation in this research which you cannot resolve with the researcher, you may contact the Ethics Committee through the Research Ethics Officer (ph: +61 2 9514 9772 Research.Ethics@uts.edu.au) and quote the UTS HREC reference number. Any complaint you make will be treated in confidence and investigated fully and you will be informed of the outcome.

Appendix 5.3(b): Interview Protocol

1. Indication of appreciation of participation and introduction (5 minutes)

First I would like to thank you for participating in this interview. Following the previous email, I have sent regarding the questions, please feel free to answer all or some of the questions. Please ask me to clarify anything, as we go through the questions. The interview should take a around 60 minutes. On completion, you can ask to:

1. Receive a copy of the final results.
2. Receive a copy of the interview transcript.

- *Ask the participants to sign the consent form*

A. Do you mind if I record the conversation? (*Start recording if yes*).

B. Do you have any question before we start?

C. I would like to start our conversation by learning something about you. Please tell me a little bit about yourself? (*Encourage participants, focus on participant's role, experience, and so on*).

2. Using agile enterprise architecture (EA) in the organisation (30 minutes)

Ok, let us first talk about using EA in your organisation. Many GDAD (scale agile) organisation, such as your organisation, have employed EA as a mechanism to enhance the communication between distributed teams (keep them on the same page) and enhance the performance of their products (projects).

1. How would you define EA in your organisation?
2. How does your organisation communicate the EA with all distributed teams?
3. Does each team have an architect (architecture owner), or someone who play the role of architect? (*If yes, what is his role?*)

Now, let us move to talk about the relation between using EA and the communication between distributed teams. Communication (active communication in this study), in general, has two dimensions: efficiency (quick, less resource), and effectiveness (clear, understandable).

4. Do you think that using EA among the distributed teams enhance communication efficiency and effectiveness? (*How if yes, why if not*)
5. Which of the communication dimension, you think, is impacted more by using EA?
6. From your experience, how using EA impact the communication with teams (team members) whose first language is not English? (*How if positive*)
7. Is this impact the same when comparing to the distributed teams inside the same country? (*Why, how if not*)

Now, let us move to talk about the relation between using EA and the performance of GDAD. Here, we refer to four aspects of project (software) performance; on-time completion, on-budget completion, functionality, and quality.

8. From your experience, how using EA in GDAD impact each of these aspects?

3. Communication efficiency and communication effectiveness (20 minutes)

Now we have talked about the role of EA in your organisation in GDAD, let us move to the second focus of this interview; the communication itself between distributed teams. As discussed before, GDAD active communication into two dimensions: efficiency and effectiveness.

1. What are the other dimensions or aspects of GDAD active communication? (*Define each*)
2. How would you describe the relationship and tension between communication

efficiency and communication effectiveness? How does the agile team strike a balance between them?

3. How does the GDAD team determine the importance/priority of communication? (e.g., decide how a particular customer requirement or change should be communicated)
4. How would you describe the tension between (1) needs for meeting time, cost, functionality and quality, and (2) needs for communicating (efficiently and effectively) customer requirements and changes? How does the agile team go about managing this tension?

4. Restating appreciation and concluding the interview (5 minutes)

1. Would you like to add any last comment?
2. Do you wish to receive a transcript of this interview? (*if yes, get the preferred contact details; email or post address*)
3. Do you wish to receive the final results? (*If yes, as above*)

Appendix 6.1: Items' Coding (Labels of Variables Used in the Study)

Construct	Variable	Label
Agile Enterprise Architecture (AEA)	Enterprise architecture framework and the framework of GDAD are aligned	AEA1
	Enterprise architecture documentations are regularly updated to align with the projects in GDAD	AEA2
	Enterprise architecture is used to define projects/programs (e.g., business/IT gap analysis) in GDAD	AEA3
	Enterprise architecture is used to assess major project investment in GDAD	AEA4
	Solution architecture, as a part of enterprise architecture, guides the projects at program levels and project levels in GDAD	AEA5
	Solution architecture evolves from small iterations, and the changes in solution architecture are reflected in enterprise architecture	AEA6
	Enterprise architecture is used to govern project implementation in GDAD	AEA7
Communication Efficiency (EFFIC)	Information needed about GDAD project is achieved quickly	EFFIC1
	Information needed about GDAD project is achieved	EFFIC2

	easily	
	The stakeholders needed to communicate with is reached quickly	EFFIC3
	The stakeholders needed to communicate with is reached easily	EFFIC4
	The cost of communication (e.g., less travelling to meet face-to-face) is decreased	EFFIC5
Communication Effectiveness (EFFECT)	All GDAD team members are clear about their tasks	EFFECT1
	Enough information is provided about customer requirements and project progress to GDAD team members	EFFECT2
	Detailed information is provided from distributed stakeholders	EFFECT3
	Accurate information is provided from distributed stakeholders	EFFECT4
On-Time Completion (TIME)	GDAD projects is completed on-time according to the original schedule	TIME1
	GDAD tams complete their tasks on-time according to the original schedule	TIME2
On-Budget Completion (BUDGT)	GDAD projects is completed on-budget according to the original budget	BUDGT1
	GDAD tams complete their tasks on-budget according to the original budget	BUDGT2
Software Functionality (FUNC)	GDAD project achieves its functional goals	FUNC1
	GDAD project meets its technical functional requirements	FUNC2
	GDAD project meets customer's functional requirements	FUNC3
Software Quality (QLTY)	GDAD project solves the given problem	QLTY1
	GDAD project improves the way of customers' use to perform their activities	QLTY2
	GDAD project achieves customer's satisfaction	QLTY3

Appendix 6.2: Correlation Matrix for Reflective Indicators (Pearson Test)

	EFFIC1	EFFIC2	EFFIC3	EFFIC4	EFFIC5	EFFECT1	EFFECT2	EFFECT3	EFFECT4	TIME1	TIME2	BUDGT1	BUDGT2	FUNC1	FUNC2	FUNC3	QLTY1	QLTY2	QLTY3
EFFIC1	1																		
EFFIC2	0.79	1																	
EFFIC3	0.82	0.75	1																
EFFIC4	0.70	0.73	0.84	1															
EFFIC5	0.57	0.61	0.69	0.73	1														
EFFECT1	0.44	0.38	0.43	0.41	0.48	1													
EFFECT2	0.31	0.38	0.35	0.29	0.39	0.77	1												
EFFECT3	0.37	0.32	0.46	0.40	0.49	0.71	0.70	1											
EFFECT4	0.39	0.33	0.46	0.39	0.45	0.74	0.72	0.87	1										
TIME1	0.61	0.57	0.59	0.57	0.57	0.35	0.31	0.34	0.4	1									
TIME2	0.62	0.58	0.57	0.55	0.59	0.36	0.37	0.30	0.37	0.90	1								
BUDGT1	0.41	0.50	0.45	0.43	0.48	0.39	0.50	0.38	0.42	0.77	0.77	1							
BUDGT2	0.44	0.50	0.47	0.47	0.50	0.45	0.47	0.36	0.42	0.78	0.79	0.90	1						
FUNC1	0.49	0.45	0.44	0.35	0.38	0.53	0.57	0.51	0.53	0.37	0.43	0.47	0.44	1					
FUNC2	0.41	0.35	0.34	0.27	0.25	0.47	0.36	0.37	0.45	0.39	0.36	0.34	0.36	0.57	1				
FUNC3	0.35	0.43	0.37	0.34	0.33	0.53	0.63	0.49	0.47	0.27	0.30	0.43	0.38	0.71	0.48	1			
QLTY1	0.37	0.40	0.40	0.34	0.31	0.52	0.53	0.56	0.45	0.25	0.20	0.30	0.24	0.43	0.46	0.62	1		
QLTY2	0.36	0.41	0.38	0.33	0.33	0.44	0.61	0.53	0.45	0.41	0.48	0.52	0.46	0.60	0.43	0.63	0.50	1	
QLTY3	0.44	0.46	0.51	0.45	0.42	0.49	0.53	0.60	0.50	0.37	0.39	0.43	0.39	0.57	0.45	0.59	0.62	0.72	1

Appendix 6.3: Non-Response Bias Test for Indicators

	F	Sig.	t-value	Sig. (2-tailed)	Mean Diff.	Std. Error Diff.
AEA1	0.749	0.388	1.891	0.061	0.453	0.240
AEA2	0.633	0.427	1.649	0.101	0.416	0.252
AEA3	0.090	0.765	1.293	0.198	0.331	0.256
AEA4	0.635	0.427	1.914	0.057	0.452	0.236
AEA5	5.398	0.021	1.442	0.151	0.370	0.257
AEA6	0.708	0.401	1.827	0.070	0.433	0.237
AEA7	0.018	0.895	1.032	0.304	0.258	0.250
EFFIC1	0.526	0.469	1.843	0.067	0.451	0.245
EFFIC2	1.712	0.193	1.621	0.107	0.397	0.245
EFFIC3	0.443	0.506	1.644	0.102	0.440	0.268
EFFIC4	0.005	0.942	2.461	0.015	0.601	0.244
EFFIC5	0.871	0.352	0.693	0.490	0.186	0.269
EFFECT1	1.553	0.215	0.229	0.819	0.058	0.256
EFFECT2	1.071	0.302	0.136	0.892	0.035	0.255
EFFECT3	0.000	0.983	0.381	0.703	0.095	0.250
EFFECT4	0.044	0.833	0.441	0.659	0.110	0.250
TIME1	0.035	0.853	0.588	0.558	0.161	0.274
TIME2	0.001	0.980	0.819	0.414	0.228	0.279
BUDGT1	0.000	0.984	0.399	0.691	0.102	0.256
BUDGT2	0.722	0.397	0.303	0.763	0.081	0.267
FUNC1	2.510	0.115	0.730	0.466	0.142	0.195
FUNC2	0.187	0.666	2.231	0.027	0.428	0.192
FUNC3	1.872	0.173	0.258	0.797	0.056	0.216
QLTY1	0.232	0.631	0.952	0.342	0.196	0.205
QLTY2	0.446	0.505	0.945	0.346	0.218	0.230
QLTY3	0.174	0.677	1.082	0.281	0.241	0.223

Appendix 7.1: Cronbach's Alpha Values for Reflective Measures

Construct	Item	Cronbach's α if Item deleted	Cronbach's α	No. of Items
EFFIC	EFFIC1	0.914	0.929	5
	EFFIC2	0.913		
	EFFIC3	0.899		
	EFFIC4	0.904		
	EFFIC5	0.932		
EFFECT	EFFECT1	0.908	0.925	4
	EFFECT2	0.912		

	EFFECT3	0.899		
	EFFECT4	0.892		
TIME	TIME1	-	0.952	2
	TIME2	-		
BUDGT	BUDGT1	-	0.950	2
	BUDGT2	-		
FUNC	FUNC1	0.648	0.812	3
	FUNC2	0.834		
	FUNC3	0.728		
QLTY	QLTY1	0.837	0.830	3
	QLTY2	0.770		
	QLTY3	0.669		

Appendix 7.2: Cross-Validation Redundancy Values for Reflective Indicators

Total	SSO	SSE	1-SSE/SSO (Q^2 Total)
EFFIC1	160.000	116.441	0.272
EFFIC2	160.000	111.624	0.302
EFFIC3	160.000	115.868	0.276
EFFIC4	160.000	122.116	0.237
EFFIC5	160.000	128.614	0.196
EFFECT1	160.000	112.055	0.300
EFFECT2	160.000	123.813	0.226
EFFECT3	160.000	120.567	0.246
EFFECT4	160.000	119.766	0.251
TIME1	160.000	89.661	0.440
TIME2	160.000	91.010	0.431
BUDGT1	160.000	103.205	0.355
BUDGT2	160.000	101.009	0.369
FUNC1	160.000	88.940	0.444
FUNC2	160.000	111.390	0.304
FUNC3	160.000	100.442	0.372
QLTY1	160.000	97.037	0.394
QLTY2	160.000	103.782	0.351
QLTY3	160.000	92.108	0.424