A Situational Agile Framework to Support Team Coordination in Global Software Engineering

By

Yaser Hafeez

Reg. No.11-FBAS/PHDSE/F09

Supervisor

Dr. Aakash Ahmad

Co-Supervisors

Prof. Dr. Sohail Asghar

Dr. Ayyaz Hussain

Department of Computer Science & Software Engineering

Faculty of Basic & Applied Sciences

International Islamic University Islamabad, Pakistan

2016

1/8/15

PhD
005.1
YAS

1. Computer Software . Development . Management

2. Software engineering

A Situational Agile Framework to Support Team Coordination in Global Software Engineering

A Thesis Presented to

Department of Computer Science & Software Engineering

Faculty of Basic & Applied Sciences

In Partial Fulfillment

of the requirement for the degree

of

PhD (Software Engineering)

By

Yaser Hafeez
Reg. No.11-FBAS/PHDSE/F09
Supervisor
Dr. Aakash Ahmad
Co-Supervisors
Prof. Dr. Sohail Asghar
Dr. Ayyaz Hussain

Department of Computer Science & Software Engineering
Faculty of Basic & Applied Sciences
International Islamic University Islamabad, Pakistan
2016

International Islamic University, Islamabad

Faculty of Basic & Applied Sciences

Department of Computer Science & Software Engineering

Dated: 31-8-2016

FINAL APPROVAL

It is certified that we have read the thesis. entitled **"A Situational Agile Framework to Support Team Coordination in Global Software Engineering"** submitted by Yaser Hafeez, Reg. No. 11-FBAS/PHDSE/F09. It is our judgment that this thesis is of sufficient standard to warrant its acceptance by the International Islamic University Islamabad for PhD Degree in Software Engineering.

THESIS EXAMINATION COMMITTEE

External Examiners:

Prof. Dr. Muhammad Afzal
Director (KICSIT). Rawalpindi. Pakistan

[illegible] 31/8/16

Dr. Wasif Nisar
Associate Professor
Comsats Wah Campus, Pakistan

[illegible]

Internal Examiner:
Prof. Dr. Muhammad Sher
Dean. Faculty of Basic & Applied Sciences.
International Islamic University
Islamabad, Pakistan

[illegible]

Supervisor:
Dr. Aakash Ahmad
Assistant Professor
NUST (SEECS).
Islamabad, Pakistan

[illegible]

Co-Supervisor:
Dr. Ayyaz Hussain
Assistant Professor (DCS&SE).
International Islamic University
Islamabad. Pakistan

[illegible]

Co-Supervisor:
Prof. Dr. Sohail Asghar
Chief Technologist
Department of Computer Science.
Comsats Institute of Information Tech.
Islamabad, Pakistan

[illegible]

International Islamic University, Islamabad

Faculty of Basic & Applied Sciences

Department of Computer Science & Software Engineering

Dated: 31-8-2-2016

FINAL APPROVAL

It is certified that we have read the thesis, entitled **"A Situational Agile Framework to Support Team Coordination in Global Software Engineering"** submitted by Yaser Hafeez, Reg. No. 11-FBAS/PHDSE/F09. It is our judgment that this thesis is of sufficient standard to warrant its acceptance by the International Islamic University Islamabad for PhD Degree in Software Engineering.

[illegible]

Head of the Department
Dr. Syed Husnain Abbas Naqvi
Department of Computer Science
& Software Engineering,
Faculty of Basic and Applied Sciences,
International Islamic University Islamabad

[illegible]

DEAN
Faculty of Basic & Applied Sciences
International Islamic University
Islamabad

Dean
Prof. Dr. Muhammad Sher
Faculty of Basic and Applied Sciences,
International Islamic University
Islamabad Pakistan

Acknowledgement:

This research journey has been challenging, rewarding and enjoyable. Though only my name appears on the cover of this thesis, many have helped me in different ways and I would like to acknowledge all who supported me to complete this PhD thesis.

First and foremost, all praises to Allah for His guidance for the completion of this work. Almighty Allah is the most generous, considerate, and compassionate who has blessed mankind with this verdict to think, explore, to learn, to discover the hidden secrets of this universe and has blessed me to get through the difficulties indulged during this study. I would also like to express my admiration towards our beloved Prophet Muhammad (PBUH) who has been a great source of inspiration of divine devotion and dedication to me; Salawat on last Prophet Muhammad (P.B.U.H).

Firstly, I would like to express my sincere gratitude to my principal co-supervisor, Professor Dr. Sohail Asghar, for his wisdom, suggestions and endless support, throughout my thesis research study. I appreciate their timely advice and interest in my work, and was always available to discuss with me. It was his efforts, courage and endeavoring attitude that helped me get through any problem at any step of this study. Without his guidance, motivation and leadership, this thesis would have never reached completion. I would also like to thank my supervisors, Dr. Aakash Ahmad and Dr. Ayyaz Hussain, for their constant encouragement. Their guidance and motivation created a momentous learning experience for me. Furthermore, I would also like to thank foreign evaluators who reviewed this thesis, especially Dr. Muhmood Niazi and Dr. Andrea Ko who helped me improve this thesis.

Specifically I would like to thank to Professor Dr. Muhammad Sher, Dean FBAS International Islamic University Pakistan, for his support, encouragement and motivation in the pursuit of my PhD degree. I am also extremely thankful to Chairman Dr. Husnain Naqvi, Department of CS&SE. for his constant guidance. I would take this opportunity to thank my faculty members, Dr. Ali Duad, Mr. Asim Munir, Mr. Muhammad Nadeem and Mr. Muhammad Imran Saeed from Department CS&SE, for their motivation. I would also like to pay my gratitude to my respected teachers; Professor Dr. Muhammad Riaz, Director National

Defense University, and Professor Dr. Arshad Ali, Director Fast University Islamabad Campus, who made me what I am today.

During this this PhD thesis, I went through many complex situations, I have had the great fortune to get to interact with Ms. Faiza. My deep appreciation and special thanks for my mentor for her prayers, endless support, constant motivation and encouragement on this journey.

I would take this moment to express my gratitude towards my colleagues, Dr. Muhammad Jamal, Dr. Muhammad Azeem, Ms. Bushra and Mrs. Shaiza , Mr. Shabbir , Mr. Muhamood, Mr. Ansar who made my candidature very enjoyable. I would also like to thank my students, Ms. Shagufta, Mr. Bilal Hashmi, Ms. Ambar, Ms. Khushbakhatwar, Ms Zartasha, Ms. Aysha, Mr. Bilal Ali and Mr. Azhar.

During this process, I went through many life events and throughout the ups and downs, my family have been giving strength and encouragement. Finally, I thank my family for the love, support and understanding during this study which is complete today because of my loving wife, son and daughter. This thesis would simply not have been possible without the ongoing support and encouragement of my parents, Mr & Mrs Abdul Hafeez, my mother and father in law, Mr. & Mrs. Muhammad Arif, my brother Hammad Hafeez, my sister Mrs. Azam & Dr. Muhammad Azam , my cousin Mohammad Omair, my brother in law, Mr. & Mrs. Zeeshan Arif, all sisters in law. I am eternally grateful to this special group of people.

The Acknowledgement section cannot be completed without thanking all those who have directly or indirectly helped me in the completion of my work. I would like to express my sincere gratitude to all those (named or otherwise) who have helped me in pursuit of this PhD degree and pray that I have the honor to walk along them throughout my life.

Declaration:

I hereby declare and affirm that this thesis neither as a whole nor as part thereof has been copied out from any source. It is further declared that I have completed this thesis entirely on the basis of my personal effort, made under the sincere guidance of my supervisors. If any part of this report is proven to be copied our or found to be a reproduction of some other, I shall stand by the consequences. No portion of the work presented in this report has been submitted in support of an application for another degree or qualification of this or any other University or Institute of learning.

Yaser Hafeez
11-FBAS/PHDSE/F09

Dedication:

This work is dedicated to my dear mother, most respected mentor, my beloved wife, my beloved daughter and my beloved son, dear mother in law, for their prayers, love and motivation. They were the light in my academic path and without them nothing of this would have been possible.

Yaser Hafeez

11-FBAS/PHDSE/F09

Abstract

Context: Modern computer systems are prone to a continuous evolution under varying situations such as Application and Organization. The increasing and complex project situations need to employ resources on a global scale that has resulted in Global Software Engineering (GSE), commonly known as Global Software Development (GSD). In addition, GSD teams work round the clock to fulfill the complex and competitive market situation to optimize and decrease costs by utilizing time zones effectively.

However, apart from the advantages in GSD, it also entails certain challenges due to distance and time difference such as information sharing, communication delays and lack of control over team coordination that lead to project failure. With the passage of time, the complexity of software has increased greatly. It is for these reasons that organizations are moving from traditional development approaches to Agile based approach as it has the ability to deliver products closer to customer's expectations.

In this study there are two main problems that have been addressed, first and foremost is problem with the integration of two popular trends in Agile practices with GSD and second is formulating a situational method in order to handle the Agile GSD situations.

However, literature review suggests the Situational Method Engineering (SME) as a solution for the integration of more than one methods and formulation of situational method. It also arises as a solution when we want to combine two approaches for the formulation of a new paradigm. Hence SME aims at providing method construction techniques by reusing existing methods, practices, techniques and approaches.

Objective: The objective of this research is to design the structure of the *3C-SAME* (*3C* _Situational Agile Methods for Global Software Engineering Environment) Framework in order to formulate situational methods to help Agile GSD practitioners to provide a productive solution.

Method: In order to design *3C-SAME* Framework, we use multi method research methodology. In addition, we use grounded theory to formulate the situational taxonomy. Action Research is also used in order to formulate the Situational Agile framework. In designing the Framework we perform a Literature Review to identify the impacts, challenges and existing issues from published articles, reports, case studies in a structured manner. In the context of the non-structured ad hoc research review, the systematic review is beneficial for the evidence based impact of existing theories, frameworks and challenges. We use the Multi method to design the *3C-SAME* Framework. We evaluate our proposed framework through ISO/IEC 9126-1 quality model, case study and experimental study.

Result: The results of the proposed *3C-SAME* Framework help Agile GSD practitioners in order to improve their performance for Agile GSD managers and practitioners. It is a viable framework which can provide solutions to the well-known problems by applying SME principles. Hence the framework brings improvements in functionality and usability of the software development processes. However, the study is limited to only two agile methods, Scrum and XP, which have been used to formulate the proposed situational framework. The proposed framework is handle specific situations not for general purpose.

Keywords: Agile Software Development, Global Software Engineering (GSE), Global Software Development (GSD), Situational Method Engineering (SME), Team Coordination.

List of Abbreviation

SME	Situational Method Engineering
GSE	Global Software Engineering
GSD	Global Software Development
DSD	Distributed Software Development
TDD	Test Driven Development
FDD	Feature Driven Development
SLR	Systematic Literature Review
SMR	Systematic Mapping Review
RE	Requirement Engineering
SDLC	Software Development Life Cycle
XP	Extreme Programming
PO	Product Owner
3C	3 (Coordination, communication, Control)
3C-SAME	*3C*_Situational Agile Method Engineering for Global Software Development Environment
CAR	Canonical Action Research
SEMDM	Software Engineering Meta Model for Development Methodologies
ISO	International Organization for Standardization
IEC	International Electro technical Commission

Table of Contents

List of Figures

List of Tables

Chapter 1: Introduction and Background

1.1 Chapter Overview

The introductory chapter describes the selection of subject matter and the situations that this dissertation addresses. It also describes the background and motivation of the research problem, reflection of research questions to attain the set objectives, research methodology and contributions to achieve solutions.

1.2 Background

Computer systems play an integral part to facilitate the human society. With the passage of time the society's dependence on computer systems and consequently their expectations may also continue to increase. The increasingly complex project situations need to employ resources on a global scale that has resulted in Global Software Engineering (GSE) becoming increasingly popular in both research and industrial perspective [1]. The notion of software development is central to this approach in which companies work in more than one location under one project. GSE is informally known as Global Software Development (GSD) paradigm in which dispersed teams work round the clock to fulfill the complex and competitive market situation to optimize and decrease costs by maximum utilization of time zones [2,3]. Today, the modern world has become a global village where GSD has gained importance, due to the leveraging of time-zone effectiveness and closer proximity to the market [4]. Furthermore, the popularity of GSD has increased due to the satisfaction level of end user. Cooperation among teams has turned out to be a significant factor in the dramatically increased growth rate [5]. It is for these reasons that more than seventy percent projects are now developed under the GSD approach [6].

However, apart from the above mentioned attraction in GSD, it also entails certain challenges which need to be overcome in order to exploit these potential advantages. GSD teams, however, face challenges due to distance and time difference during development, such as cultural diversity, language difference, information sharing, delay in communication, lack of control over the team and team coordination that leads to project failure [7, 8]. The competence and capabilities of the participants in the distributed development, the available resources and the organizational environment is vital in deciding the success or failure of distributed projects [8, 9]. Lack of project management support, lack of standardized practices in software development and lack of team coordination, control and communication is unfortunately common. Furthermore, little customization of frameworks, lack of requirement change management and poor knowledge management are listed as recognized problems in software development [10].

The software industry has grown tremendously in the last few decades. With the passage of time, the complexity of software has increased greatly and its understandability is almost impossible for individuals. In order to reduce the complexity, group work is gaining more importance. According to Standish Report, more than 75% of the participants respond that their organizations have implemented agile development practices [11]. It is for these reasons organizations are moving from traditional development approaches to agile based approach as it has the ability to deliver products close to customer's expectations as compared to the traditional method [12]. This multifaceted growth can be attributed in order to capitalize on the global resource pool to fulfill the challenges of the modern era. Moreover, they are looking for an appropriate mix of expertise to fulfill the objectives of cost and time reduction [13]. Hence, the most popular trend adopted by the distributed teams is the implementation of agile methods, despite the separation of team members through space, time and culture in order to achieve a good impact on software production [14]. It is vital to recognize the overall integration of the two popular trends in order to benefit from combining agile methods with GSD.

Despite their significance, representations are not the only success factor in software development process. There are numerous studies [15] showing that there is a dire need for an in-depth study to determine how distributed teams effectively use Agile methods while time, space and culture constraints still exist. Smite et al. [16] argues that there is still no strong consensus for the combination of Agile applicability with distributed development to get expected benefits of an integrated approach. Later on, Smite et al. [18] suggested about the future research trends that a theoretical model with the combinations of both Agile and global software development must be formulated to get the benefits of two popular approaches. Jalali and Wohlin [16,41] argue that the majority of the studies suggested that the future for Agile development approach in GSD environment research needs to be rigorous research dimension.

It is obvious that one single software development method cannot cover all the aspects, but different success factors and viewpoints need to be considered in an approach which supports a flexible combination of analysis, design and management conventions. Thus supporting different best practices in the distributed environment as well as integration of Agile software development approaches is an important topic. [19] Despite the research effort during this decade, exploring new software development methods for complex situations is still a valid research goal for situational method development for globally distributed development.

Problems vary from situation to situation and are characterized on the basis of contextual factors. It is generally acknowledged that it is not feasible to have a universal method without any modifications used in all situations [20]. The Situational Method Engineering (SME) matures as a reaction to the above mentioned problem. It also arises as a solution when we want to combine two approaches in order to formulate a new paradigm. Hence SME aims at providing method construction techniques by reusing existing methods, practices, techniques and approaches [21]. The new developed methods are more flexible and better adapted according to the distributed situations. The overview of Agile GSD challenges [10...16] is shown in Figure 1.1.

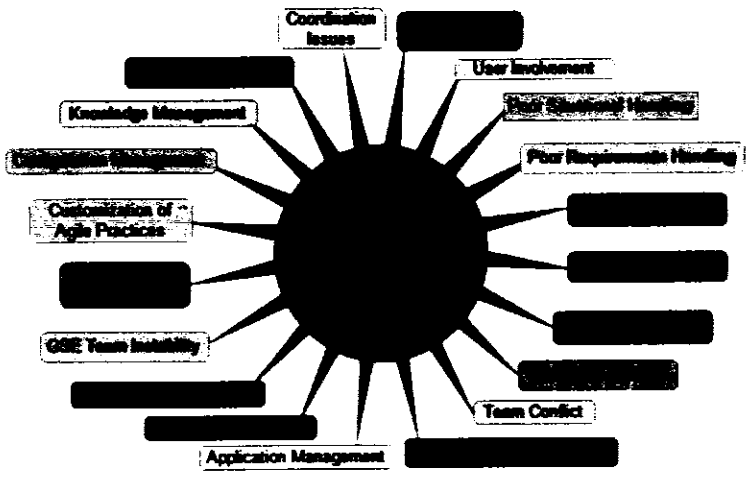


Figure 1.1: Overview of Agile GSD Challenges

It is for these reasons that this dissertation attempts to provide framework that provide the solution to the above mentioned problems.

1.3 Problem Statement

Participants in GSD projects interact with people from multisite locations and diverse backgrounds, having lack of domain knowledge and lack of team coordination, and thus signify dispersed teams confronting with different complex situations [23]. However, the subsequent effects of poor handling of situations regarding organization, project and application comprise inflated rework, delay in schedule, poor quality projects, unhappy client, and system failure [24]. But despite the obvious need for a contextual approach in order to handle the situational challenges, there remains an important gap between software development theory and practice.

On the other hand agile methods are now perceived as the universal remedy for software development project failure [25], but its adoption in the GSD still faces problems e.g. many GSD organizations wanting to adopt agile methodologies are finding difficulties in selecting a particular agile method from the agile family in a distributed environment [26]. Even after the selection of a particular Agile method, they still find it difficult to customize it according to their organization, application and project specific situation. Hence, there is lack of logical approaches with the situational process, regulation and an appropriate way of selecting and customizing agile methodologies so that organizations can select and tune it according to their specific needs and situations to support team coordination [22].

As a result, an appropriate method to handle situations is an issue of paramount importance in agile methodologies under distributed development to fulfill the gap between research and practice. Furthermore, there is a dire need for continuous improvement in the execution of existing software development process, and the results it produces.

1.4 Research Objectives

The main objective of the thesis is to develop a situational framework based approach for the GSD environment of an agile based organization. We focus on the following objectives:

Obj1: The main objective is to critically analyze prevailing state of the art in Agile GSD, including the existing taxonomies, approaches, models and frameworks.

Obj2: Investigate the available challenges in GSD as reported in the literature to formulate an appropriate situational taxonomy to classify the possible GSD challenges in a hierarchical order.

Obj3: Design a Situation Specific Framework based approach to support practitioners to formulate situational methods for Agile GSD development.

Obj4: To evaluate the situational framework in terms of improvements to usability and effectiveness.

1.5 Research Questions

In order to fulfill the objectives, we divide the research tasks into a set of research questions. Each research question (RQ) helps to tackle research problems of the proposed research.

RQ 1: What are outcomes of the existing literature about the Global software engineering possible challenges and their possible classification?

RQ 2: What types of Agile solutions have been suitable for GSD and their impact on GSD Challenges?

RQ 3: What is the state of key components of the situational frameworks which have been provided to situational methods as solutions?

RQ 4: What are the Agile GSD situations that have been exploited for a solution?

RQ 5: What evaluation methods have been used to validate the proposed solution?

1.6 Research Scope

Although the Agile software development in GSD environment has received significant degree of attention in the existing related studies to date, there still remains a justified need for new methods that can be easily embodied and enhanced by the majority of practitioners. In the current study, we argue that the majority of the studies suggested that the future for Agile development approach in GSD environment research needs to be rigorous research dimension.

The principle focus of this study is on the integration of Agile methods with GSD environment, and not the other software development methods related to analysis and design such as waterfall. Furthermore, we concentrated on only two Agile methods, scrum and XP for fragment management. It is during this preliminary phase that a situational method would be generated that is suitable for existing practitioners.

The scope of the current study focuses on combining Agile methodologies with GSD environment. As highlighted above, even after the selection of a particular Agile method, practitioners find it difficult to customize it according to the need of their organization. Hence the combination of these two methods is important so that situations can be handled more efficiently under distributed development to fulfill the gap between research and practice. The evaluation of the proposed framework is mainly performed for ISO based model, case study and experimental study.

1.7 Research Methodology

We follow the guideline of multi-method research methodologies that provide a rigorous set of research steps. We consider this methodology in order to get the benefits of both quantitative and qualitative methods. Furthermore, detailed description of both methods is given below.

Quantitative research studies comprise of natural phenomena applied through experimentation and hypothesis testing, and thus measures and evaluates instrumental relationships between variables within the phenomena under study [97]. Hence, quantitative approach assembles statistical data from a demonstrative sample and evaluates them, usually through experimentation and quantitates methods. The main objective set forth by quantitative methods is evaluating the relationship between independent and dependent variables, and exclude inappropriate variables, therefore making initial hypothesis acceptable or reject able. As a result the quantitative approach primarily encourages experimentation and hypothesis testing Grounded Theory [30] in order to propose a situational method.

Qualitative research represents study addressing social and cultural experiences through direct interaction with individuals' involved within the study. Hence, qualitative research comprises of data gathered through interviews, documentation and observations. Most data acquired is represented in the textual data. Furthermore, it covers social issues related to social and communication behaviors. Also, qualitative methods support quantitative methods by forming the data inputs for experimentation and statistics.

However, Multi-method is applied by incorporating different research methodologies for gathering and analyzing data [27]. When choosing specific research methodologies, usually one research method does not satisfy the reliability [29]. Multi method research overcomes the barrier of incorrect conclusions obtained through single research method [28]. Multi method is identified as the solution of single study research problems. Obviously, it is a superset of the earlier approaches; and potentially lessens a lot of the confines of the prior "solutions" [27]. Multi-method allows inclusion of different characteristics of other methods such as questionnaires, interviews, experiments [27]. The multi-method approach uses a mix of empirical research methods, as contrasted to single research methods, as an approach for attaining reliable research results, thus overcoming biasness [32].

The research strategy used in this thesis involves different research methodologies, thus comprising them under the umbrella of multi-method research methodology. It is vital to mention that use of empirical methods is vital, since they allow the practitioners to incorporate multi-disciplinary and interdisciplinary factors that commonly arise, such as social issues, communication barriers, situational factors and product quality. Furthermore, the research methodology is a science to study how research can be conducted scientifically [33]. Systematic conduction of research in software engineering requires deep understanding of available research methods and how their research problems need to be applied [34]. A number of research methods are recognized to support software engineering research using multi-method methodology [35].

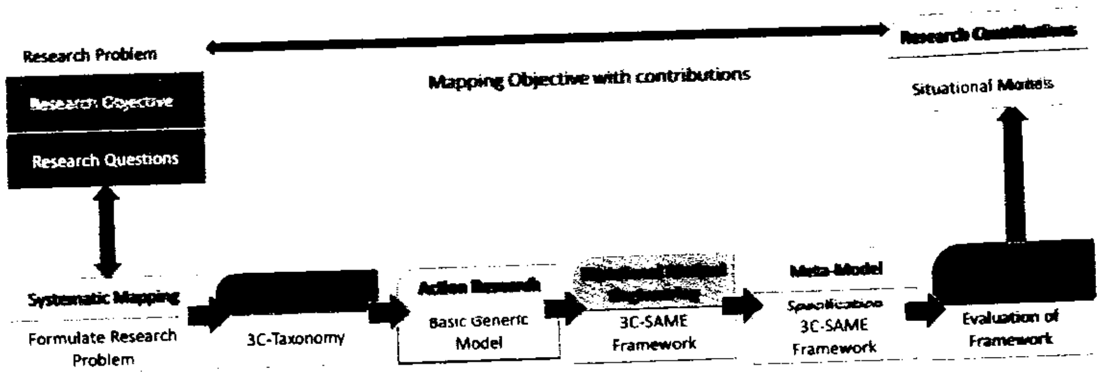


Figure 1.2: Conducted Multi Method Research Methodology

In the Figure 1.2 different research methods are described which are conducted for this study. The proposed methodology has been involved through problem formulation. Firstly, a research area was selected, then research objectives were set and then research question were formulated to fulfill these objectives. The systematic mapping study was conducted (See Chapter 2) for defining and describing the problem. Similarly, context problems were also refined. Participants were consulted for publishing reports and data gathering. They have Agile GSD basic knowledge. Grounded theory was applied for extracting codes on the bases of defined issues and proposing theory in order to formulate situational taxonomy based on identified challenges, in order to provide a guideline. Most critical challenges were identified and interpreted with the help coding scheme (See Chapter 3). This phase is formally named as solution space in order to formulate Basic Generic Model that will be formulated with the help of action research method (See Chapter 4). We propose a Situation Specific *3C-SAME* Framework with the help of situational method engineering approach which provides a base to formulate situational method into work. In the evaluation of part we use real world case studies and experimental study to evaluate our *3C-SAME* framework into work in real phenomena. At the end of the study, the contributions is also mapped with the set objective (See Chapter 7) which is defined earlier.

1.8 Research Contributions

The research presented in this dissertation is a valuable addition to the body of knowledge in the field of Agile GSD:

1. We perform critical analysis and review with the current state of art, including the available taxonomies, frameworks, approaches, and models.
2. We propose a *3C* situational taxonomy to facilitate the GSD practitioners in order to classify the identified challenges.
3. The Proposed approach is formally known as a *3C-SAME* framework (*3C* _Situational Agile Methods for Global Software Engineering Environment) that utilizes the SME principles.
4. We present a *3C-SAME* framework into work that utilizes, and generates a successful method. The newly proposed framework extends support to handle the situations for Agile teams in a distributed environment in order to shift theory into practice. It has also embodied and enhanced the*3C-SAME* Architecture with Meta model.
5. The effectiveness of the proposed approach is evaluated based on its validation methods such as the ISO 9126 model, experimental study and empirical case studies.

1.9 Outline of the Thesis

The structure of the thesis is described as given below.

Chapter 1: Introduction

The primary goal of this chapter is to provide the foundation of the study and present the overview of the rest of the dissertation. In this chapter, we introduced the research motivation and background with a problem statement, including the research objective, research questions, research methodology, and research contributions.

Chapter 2: Literature Review

In this chapter, we review the existing related available literature about subject matter, including the existing taxonomies, frameworks, approaches, and tools. We present the steps of mapping study, conducting mapping process and data extraction strategy. We also describe the global software development challenges, existing Methodologies to mitigate these challenges, outcome of literature and broader overview of the proposed solution.

Chapter 3: Proposed *3C* Situational Taxonomy

In this chapter, we proposed a *3C* taxonomy for global software development teams. We present overview of GSD challenges, analysis of existing taxonomies, research methodology, taxonomy formulation, empirical validation of taxonomy and discussion of practical implications.

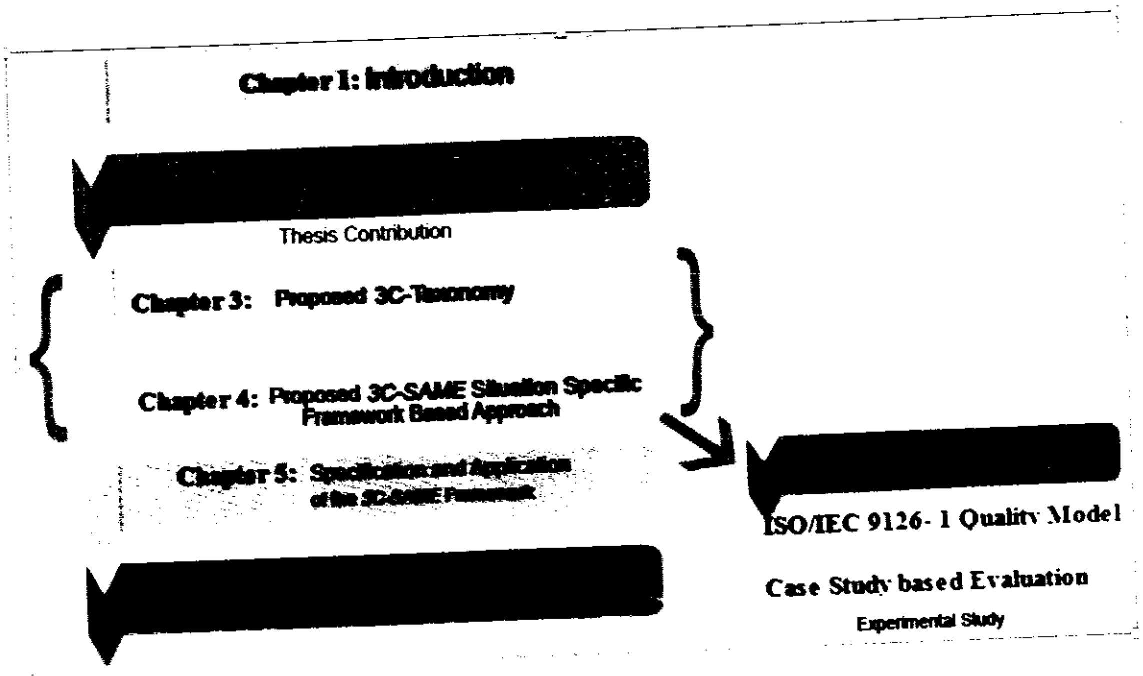

Chapter 1: Introduction
Thesis Contribution
Chapter 3: Proposed 3C-Taxonomy
Chapter 4: Proposed 3C-SAME Situation Specific Framework Based Approach
Chapter 5: Specification and Application of the 3C-SAME Framework
ISO/IEC 9126- 1 Quality Model
Case Study based Evaluation
Experimental Study

Figure 1.3: Thesis Outline

Chapter 4: The Situation Specific *3C-SAME* Framework Based Approach

In this chapter, we overview the proposed approach and then background and motivation to adopt (SME) for the development of our proposed *3C-SAME* approach. The main purpose of this dissertation is to provide an innovative and enriched approach for practitioners and *3C-SAME* framework components and its capabilities. In the last section we evaluate our proposed approach with ISO 9126 standard.

Chapter 5: Specification and Application of *3C-SAME* Framework

In this chapter, we present the overview of framework, the background, *3C-SAME* architecture, prototype specification and its artifacts. Furthermore, we compare and contrast our proposed approach to existing related work role of Meta model.

Chapter 6: Evaluations of *3C-SAME* Framework

This chapter, presents the evaluation overview, Experimental Study, ISO based model, background for analysis with a case study, case description, case study execution procedure and evaluation results. The objective of this chapter is to empirically evaluate our proposed *3C-SAME* framework into work and to determine its artifacts with relative usability and effectiveness.

Chapter 7: Conclusion and Future Work

We illustrate the conclusions, findings, significance and suggestions for future research directions.

Chapter 2. Literature Review

2.1 Chapter Overview

In the previous chapter, we presented the introductory building blocks of the dissertation. We defined research motivation and background, with respect to Situational Agile development in a distributed Environment. We then defined the set of research objectives and questions, a statement of the problem, research methodology, contributions and outline of the thesis.

The primary goal of this chapter is to provide a critical literature review about the fundamental concepts in and around the topic of situational Agile framework in the globally distributed environment as shown in Figure 2.1. Furthermore, this section illustrates a literature review conducted before the analysis of the individual research questions outlined in chapter one.

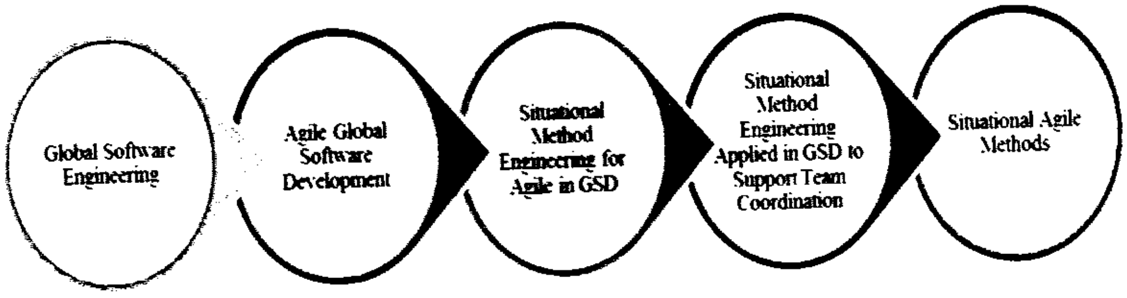

Global Software Engineering
Agile Global Software Development
Situational Method Engineering for Agile in GSD
Situational Method Engineering Applied in GSD to Support Team Coordination
Situational Agile Methods

Figure 2.1: Topic Dealt with literature Review

Therefore, we perform critical analysis of previous research and consider various concepts and the subsequent theories that are relevant to our research domain and which need an upfront explanation before the technical solutions are described.

We adopted Systematic Mapping Review (SMR) to analyze the problems and their possible solutions [36]. In order to gain a broad and in depth understanding about topics, SMR is applied to the appropriate literature review technique in an Agile GSD environment to mitigate the team coordination situational challenges which result in the identification of potential research gaps within existing related studies. We, then investigate with the goal of identifying unresolved issues that require further investigation. Main focus here is on the problems met during global software development due to inappropriate selection of Agile practices. This study reveals that the use of a working prototype has a significant impact on customization and tailoring of agile methods. We mainly concentrated on clarifying issues related to combination of GSD with Agile practices and then achieving situational Agile methods in the rest of the chapter.

2.2 Overview of the Systematic Mapping Review

In this section, we briefly introduce the method of investigation for literature review, the Systematic Mapping Review (SMR). In the current era, the systematic review process is popular among the research community, due to its specific and precise information gathering style [36]. Furthermore, the researcher's trend has dramatically changed from unstructured literature review, towards systematic literature review. SMR is also designed to give an overview of a research area through classification and rigorous sequencing of methodological steps [34, 37]. Furthermore, it is helpful to identify, evaluate and interpret all the presented interrelated studies. Evidence based research has been popular not only in software engineering discipline but in other domains as well.

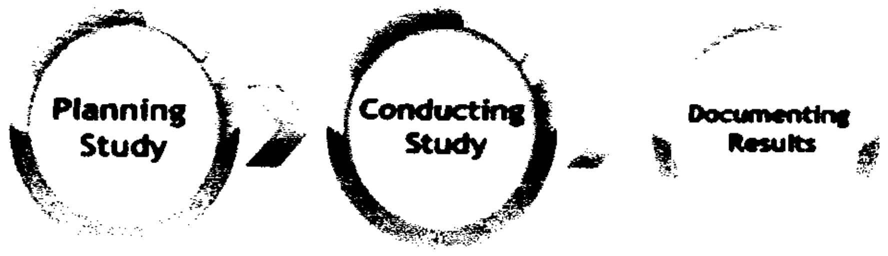


Figure 2.2: Overview of the Systematic Mapping Review

We carried out SMR to identify and analyze potential challenges of previous GSD related studies. We synthesized the data from selected papers to (a) Identify the list of GSD domain challenges and subsequently classify the existing research gaps and (b) systematically combine GSD with Agile solutions. Furthermore, it is helpful to provide background for Agile GSD and formulate new solutions of research by applying Situational Method Engineering (SME). Subsequently, we present the main steps for conducting systematic mapping i.e. planning the study, conducting the study, and documenting the results[52]. Figure 2.2 shows the overview of systematic mapping review.

2.2.1 Planning the Study

The first and foremost phase of SMR is the planning phase, which consists of identifying the need of review, research questions (See Chapter 1) and review protocol respectively.

2.2.2 Conduction Review

In this phase of SMR, we focus on inclusion and exclusion criteria to analyze the related literature. Relevant parts of the study were extracted and the results concluded as shown in Figure 2.3. Keywords and search strategy helped to find the available material specific to the research objective.

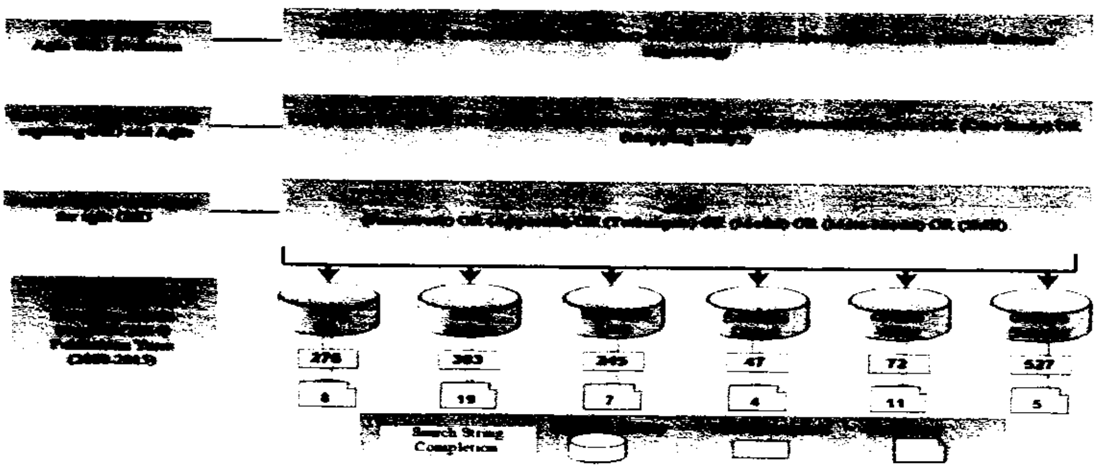


Figure 2.3: A Summary of the Primary Search Process

The strategy involved scoping the search space which comprised of electronic libraries and databases as shown in Figure 2.3. The studies were initially retrieved through electronic library and databases during 2000 to 2015.Furthermore, the protocol of the study is attached at Appendix G. Outcome comprises of framework based approach, which may be shifted into a working prototype in order to shift theory into practice. The Table 2.1 has shown the generic and documentation specific data about the study.

Table 2.1: A Summary of the Extracted Data and Comparison Attributes

[illegible]	[illegible]	[illegible]
1	Study ID	Unique ID of Study
2	**Bibliography**	**List of Authors, Year of Publication, Source of Publication** ***[Books, Journal, Conference, Workshop and Others]***
3	Focus of Study	Theme, Concepts and Motivation clearly presented.
4	Research Method	[Grounded Theory, Case Study and Action Research]
5	Research Problem	Customization and Producing Situational Methods for Agile GSD Practices.
6	**Proposed** Solution	***Situation Specific SC-SAME*** **Framework Based** Approach
7	Application Context	Context and application domain: [Theory and Industrial Practices]
8	Evaluation **Method**	[ISO 9126 Model, Case Study ,Experimental Study]
9	Future Dimensions	Findings and Future Research clearly stated.

2.2.3 Results of Review

In this section, we carried out the results of the literature review. We arrange our results into further subsequent sections. In Section 2.3, we present only Global Software Development (GSD), Environment related studies, as our scope of study is working in a distributed environment and global teams distribute their tasks to get benefits from skill and cheap labor across the world and use time zones optimally. Then in Section 2.4, we present Agile practices in a GSD environment to get benefits with the integration of two popular paradigms. After this, we present applying Situational Method Engineering (SME) guidelines to further customize the Agile GSD Approach according to project, organization and Application situation in Section 2.5. In this context, we only present brief description of literature review results in the following manner:

- A general description of the study.
- The research contribution.
- At the end of each section list of key findings of study, if any.

2.3 Study of Global Software Development Environment

In this section, we mainly focus on the most relevant methods in Global Software Development (GSD), paying special attention to how GSD is specified in a situation support coordinated action during distributed development Environment. The most important and relevant studies under GSD are given below.

For the last twenty years, a concept of organizing projects in geographically distributed environment has evolved and gained immense research attention. Thus, due to complex project situations, employees need to be resourced on a global scale [23]. The situation in which different teams work together to achieve common goals without considering the borders of a nation is called Global Software Development (GSD) [23]. Due to the geographical distance in time and space between teams, communication and information co-ordination among team members remains a problem [17].

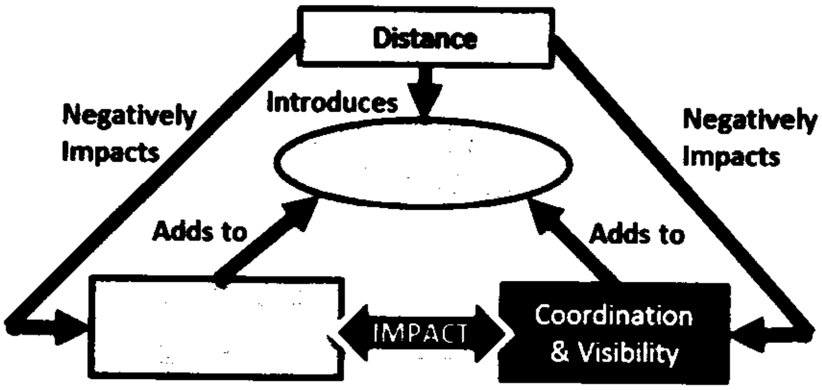


Figure 2.4: Impact of Distance on GSE Teams [17]

The key idea in geographical distance is that multisite development affects the communication process and hinders team coordination [16, 23]. Figure 2.4 highlights the issue of distance that plays a negative impact on communication and team Coordination.

Today computer organizations need to use their existing resources as efficiently as possible. They also need to centralize their resources on a global scale from different sites. This has resulted in Global software Engineering (GSE) that has the capability to cooperate with different teams over geographical and temporal distances, consequently concluding in global distance. Thus, the teams working under GSE settings are known as global teams [39].

˘Smite et al. [18] focused on empirical studies and therefore, they provided Systematic Literature Review (SLR) in order to get understanding of the GSE domain in depth. They identified useful practices and techniques along with potential benefits and constraints. Furthermore, they described from related literature, the seven most popular practices used by GSE practitioners. At the end they provided the strength of empirical evidences, sources of evidences, principle findings and implications of the study.

De Silva et al. [39] presented a Framework for Global Teams to get benefits of Software Product Line. They focused on requirement engineering activities, such as elicitation, analysis and requirement management. They also highlighted the ignorance to adopt the transitional techniques such as requirement tracking repositories. This research focuses on the solution perspective of designing and construction of the product line on GSD environment.

Naizi et al. [40] presented a Systematic Literature Review (SLR) about GSD project risks. They highlighted reasons for project failure, such as the complex and versatile nature of project. They identified approximately 37 papers of GSD domain and extracted required results from about 24 of them. The main conclusion of the study was to provide optimal solutions for GSD risks as the nature of the project and their situational factors vary from project to project. Yet, there were several types of risks discovered, categorized as requirement engineering risks, software development risks, architectural risks, communication risks, control risks and coordination risks. The authors believed that there is need to mitigate the identified risks and for better solutions. The most frequent challenges, according to them, are the lack of team coordination and communication.

Rodríguez et al. [38] performed a Systematic Mapping Review (SMR) in GSE context. They focused on GSE tool, working under highly dispersed situations to achieve the common goal. They focused only on coordination and communication tools to support global teams. In this study, they discovered all the available communication and coordination tools to support global teams under remote locations. They described and then classified GSE tools in such a way that both practitioners and researchers can use their study in daily decision making and collaboration. They identified approximately 132 tools of GSE and their usage for commercial, free or research oriented projects. The main intention of these tools was to mitigate the impact of lack of communication, coordination and control.

Richardson et al. [53] presented a framework to support GSE teams in a distributed environment. As a result, the software manager faced a different problem to manage and coordinate the global teams in varying situations. The research contribution is to present a process area to address specific problems related to geographic and linguistic distance and to fulfil the complex needs of global teams. They carried out three case studies to evaluate their proposed work. Furthermore, they identified a few threats for the guidance of software managers on tackling the situational challenges during GSE Projects.

Beechom et al. [43] presented possible solutions of existing highlighted GSD challenges. Previously we have seen many empirical studies to identify the GSD challenges, but this study collected the mitigated solutions empirical validation about the identified solutions. At the starting point of this study, they presented the GSE model, and then categorized the possible solutions into process areas. For this review they identified approximately 330 papers of GSD domain, extracted 127 solutions and then mapped them to the GSD model.

Christopher et al. [45] highlighted the most critical GSE risks, and analyzed the impact of these risks in GSD projects. They suggested a few actions to mitigate the relevant GSD risks. They carried out case study approach to evaluate the proposed solution, and involved 15 cases from seven different GSD industries for industrial evaluation. However, this study only deals with the generic GSD risks and does not cover the risks specifically related to team coordination, control and communication.

Šmite et al. [46] proposed a taxonomy to support GSE practitioners. Many organizations shifted their traditional development environment towards GSE due to competitive market situations and thus GSE strategies were discussed more often to get maximum benefits from global teams using high skill and cheap labor. With time, many terms and jargons have been introduced in GSE. However, this variation in terms, also hinders the understanding of the semantics of GSE terms during the selection of the appropriate terms under the varied situations during GSE development. So there was a dire need for accurate terminology to accommodate for different GSE situations to facilitate GSE community. In this study, they proposed the empirically based taxonomy to facilitate the researchers and practitioners.

Nuevo et al., [47] compared to the relatively recent emergence of GSD literature shows that there are many evidences available from experience reports and case studies which illustrate that team coordination and customization issues require serious attention in multi-user environments. But despite the obvious need of an appropriate method to overcome the geographical distance, face-to-face communication is a complex task, and tailoring and customization is still a gap between theory and practice. To mitigate these issues of the modern era, practitioners are looking for a comprehensive approach to minimize the gap between academic and industrial practices. Although GSD could be made more successful and productive during the development cycle, but the obvious need for contextual approach which may handle these issues might make a significant contribution to proposed situational methods.

In summarizing, the cited issues described by numerous studies above can affect the project quality that results in unsatisfied clients. Measures should be taken to manage and mitigate these issues to improve the project development process. In this section, the key findings about the GSD studies were identified which are given below:

- Global distance is the root cause of different challenges such as lack of team coordination and lack of communication between global teams [47].
- To overcome GSE challenges in dispersed situations, systematic approach is required [18].
- Different background and inappropriate project Organizations trouble project productions and software quality [48].
- Unclear requirements and lack of team coordination may cause project failure [39].
- Lack of appropriate knowledge repository may cause issues such as shared understanding for GSD teams [45].

2.4 Study of Agile Global Software Development Methods

This section emphasizes on applying Agile software development practices in Global Distributed Environment, which provides a foundation for developing the proposed framework. The methods presented above (See Section 2.3) are mainly focused on providing mechanisms for globally distributed teams working under global distance, which produces challenges such as coordination, communication and control. Currently, software managers choose best practices to perform customization and tailoring based on their personal experience. The result is project failure. To overcome and reduce the project failure rate, practitioners look for logical approaches with situational process regulation.

Further analysis of literature on Agile practices with GSD environment also provides a useful contribution for practitioners in two aspects: First aspect, is to provide the possible solutions from agile practices to mitigate GSD challenges (if any), the second aspect is to customize and tailor Agile with GSD. The brief description of Agile GSD methods is given below for further understanding.

Hossain et al. [48] characterized a framework using agile practices to overcome GSD challenges such as communication, coordination and control. It is a conceptual work to judge the effectiveness of agile strategies. In addition various issues such as global distance and global teams were identified. They focused on two extremely popular approaches, first is Agile practices and other one is GSD environment, both of which are combined to leverage the benefits of modern development approach such as Agile, over traditional development approach such as waterfall. They identified a few challenges such as communication and coordination which can be mitigated with the help of applying daily scrum from agile methods. Their theoretical framework also mapped Agile scrum practices with GSD challenges. In this review 400 papers were collected but results were extracted only from 20 to provide the conceptual framework.

Bannerman et al. [49] provided a study to overcome the GSD team coordination challenge by applying Agile Scrum method. They performed four empirical case studies, from different projects to leverage benefits of Agile/Scrum practices under GSD context. Their research contribution is that empirical validation of Agile/Scrum method can be beneficial in a GSD environment. However, previous traditional development methods struggle to compete with current needs of the modern era.

Qahtani et al, [50] proposed a Framework for Applying Agile practices for Software Products. They focused on a variety of challenges related to the technical aspects like communication, configuration and integration management of distributed development, but they ignored social issues such as language barriers between distributed teams. This framework was developed to customize an agile approach for software product during GSD environment.

Paasivaara et al. [6] presented a solution for product owners for Agile Global Teams in Software Product which stimulate the individual's critical and creative role for team collaboration. This would ultimately address and provide a consolidated and detailed specification of technical and nontechnical attributes and formal communication in Agile teams as well as informal communication in product management, where the technical attributes include communication and non-technical attributes include improved collaboration for the software product.

Petteri et al. [51] described GSD issues such as cultural, time zone, communication etc. They focused on the effective usage of agile values along with principles have great impact on communication. For the empirical validation of their problem, they spent more than three months for collection of empirical results and involved 25 participants in GSD environment. They used version control system (VCS) to analyze the data between the participants. In addition to pre course survey, sprint survey, meeting, video and observation are used for data collection between multi-site teams. They described that the iterative development makes simple for stakeholders to communicate with each other. They recommended that there is a need to facilitate awareness to the distributed teams during the collaborative software development activities.

Shrivastava et al. [44] described the potential benefits of combining Agile practices with GSD and presented a few proven practices to mitigate the identified GSD issues. They presented various types of factors (such as pool of global resources and pressure to adjust time zone differences), which may be helpful for situation management. The different challenges faced by Agile GSD teams include different working hours, lack of rich conversation, distribution of work and training on Agile practices. They suggested a few tools for the support of Agile GSD teams and categorized them with respect to their functionality, such as social networking tools, communication tools and knowledge centers. The conclusion of this study is that combining Agile with GSD has a lot of benefits and even the vital issues such as communication and coordination may be minimized up to some extent.

Tariq et al. [42] designed a Framework supporting team activities both technical (included requirement for retirement) and managerial (included project planning to project monitoring) for distributed projects. In addition, a few tools like language translator and content version system were also introduced. In particular, the activity and process aspects of this framework is notable. In spite of this, few issues still required attention, like reducing costs and time to market for dispersed teams. The same idea is argued during the designing of our proposed framework.

Quershi et al. [54] has proposed mitigating coordination costs in GSD Using Scrum. The behavior and active structure elements are being applied to support a unique feature to organize local and global scrum meetings. In particular, this activity may help to reduce the coordination cost to mitigate the effect of socio-cultural and geographical distance. The framework is a conceptual proposition which requires empirical validation with real world case studies. Their proposed solution also provides a meeting schedule to reduce the coordination costs challenge for GSD projects.

Pocatilu et al. [57] provides a cloud computing paradigm which is being applied to resolve GSD Challenges. Their contribution is towards cloud computing features and privileges both in terms of product and process for the uplifting of distributed development. Web services have been used for the transmission of knowledge for the cloud.

From the promises and conclusions of this work, we draw motivation for our efforts to maintain the central knowledge factory which includes cases to construct the repository and it may also be applied for comparison with earlier experienced cases.

The Zachman framework [66] is an enterprise ontology which comprises of comprehensive classification names, model names, enterprise names and audience perspectives which can be applied to focus the designs and transformations on specific viewpoints, aspects and levels of abstraction. In this process, responsibility, timing and motivation aspects are most directly related to the activity and business aspect of this study. This framework provides only mapping with classification name and audience perspective and did not provide any conceptual and logical workflow.

Yildiz et al. [56] introduced a framework for Application Architecture which is relatively a unique approach and has received significant attention for inhibiting communication and coordination for GSD teams. They developed a tool which can also characterize the declared lists of GSD challenges. Their contribution included a Meta model and a process for deriving application architecture. At the end they developed a global architect tool for proof of concept of the framework. They applied case studies to empirically validate their tool.

Agile practices involve care adoption to distributed situations [60]. Another effort by Babar et al. [48] represents ten success stories about successfully launching of agile GSD projects. The limitation of this study is that it did not illustrate about the selection of an appropriate agile method in GSD setting. On the other hand, Hossain et al. [14] proposed a conceptual framework for globally distributed projects using an empirical project to mitigate the GSD challenges. This conceptual framework is silent about the implementation of proposed work and it is much more difficult for the practitioner to get benefits from conceptual framework. Another study in which, authors presented [40] a systematic literature review with the primary intent of solutions towards GSE challenges provides an optimal solution to improve software development in global software engineering but gives no attention towards handling of versatile situations [21].

Given the growing attention, research analyst reported that for timely and cost-effective release of quality software, GSD practitioners use iterative software development methodologies [62]. Thus, agile methodologies are best suited to be adopted by GSD organizations [63]. They suggested a distributed Agile development and an innovative approach for mitigating the effects of GSD challenges like; time zone differences, diverse location and cultural issues. Hence, there is a dire need to provide a comprehensive solution which provides a new paradigm for agile distributed teams in order to overcome the versatile situations. Only then can Agile practices be used to mitigate GSD challenges.

2.5 Study of Adoption of Situational Method Engineering (SME) in Agile GSD Methods

We argue that when the organization structure varies from small to large size, we need to apply Situational Method Engineering (SME) to cater to the issue of size and application in different situations. Since the last decade, several studies were conducted related to the agile method customization using SME. Some of them are presented as under:

There are numerous well known agile methods such as Scrum, XP and Lean. But these methods do not fulfill the particular needs of different situations during the distributed project development [64]. They provided a basic ground for one of most common approaches called assembly-based SME approach to develop and design the agile methods. For this purpose, they identified a large set of components from the existing agile methodologies. It fulfills a wide variety of requirements of method engineers who are responsible for developing new methodologies [55]. They have proposed an agile method base which consists of the necessary components to satisfy different situations of the project. The agile method base components proposed, conform to the Software Process Engineering Meta model (SPEM) standardized by the Object Management Group (OMG) [21]. It has also been plugged with any Computer Assisted Method Engineering Tools which implements SPEM Meta model [68].

There are a variety of agile methodologies for software development, but there is no appropriate method for customization and adjustment of agile methods. Van-De-Weerd et al. [67] developed a framework which helps in adapting agile methodologies up to some extent. They also presented a Meta model for their proposed work. Their Meta model was derived from both direct and indirect solutions of agile method. However, they failed to provide any solution to the team coordination issues in Agile GSD.

Ayed et al. [65] developed a Meta model which is specifically designed to support the customization and construction of agile methodologies along with different measurements. It also provides guidance in enacted agile method based on these measurements and provides feedback to the agile methodologists and method engineers so that they can adjust during development phase.

Even though there are popular and very well-known Agile practices, but most of the methods are not ideally suitable for different situations [88]. Consequently, it is necessary to use SME to construct agile methods in GSD to fulfill the needs of software project situations [88]. To achieve the quality objectives of software development, practitioners suggest modification in agile methodologies for particular situations [21]. As the consequence of this need, we investigated different approaches from above mentioned solutions which are helpful for outcome of the literature.

2.6 Outcome of the Literature Review

In this section, we are further defining and describing the outcome of the literature. The purpose of this section is therefore to define outcome key concepts about the main topic we have addressed, and consequently lay the foundation for the rest of the thesis by clarifying the scope of research. It is vital to mention about identifying challenges that covered perspective, theory and industrial practices. In this context, the existing efforts mainly focus on introducing or integrating solutions, Agile practices with the GSD, up to some extent, but as time progresses, every solution has some limitations, particularly when we combine GSD with agile new issues are highlighted, and this requires attention. The basic objective of identifying and addressing the fundamental problems for highlighting challenges is pinpointing the research needs. We are focusing on only those findings of our literature which required immediate attention and draw following conclusions.

- **Conclusion 1: - Team Coordination**: Due to the geographical distance in time and space between virtual teams, information co-ordination becomes difficult. Currently literary articles cited (See Section 2.3) earlier highlight the Coordination problems, for distributed settings. There is a growing literature that demands more empirical study to understand the issue for team coordination in Global Software Engineering.

- **Conclusion 2: -Combining Agile Practices with GSD:** The use of Agile practices may be limited in Global Software Development. The literature evidence clearly supports that agility works for all small, medium as well as large size projects but global teams face different challenges. Our review revealed whether or not Agile practices can successfully be used in GSD setting. Agile promotes communication and interaction among the global teams and thus team coordination improves [65]. Agile Manifesto and its underlying principles do not argue against the feasibility of method composition factors according to its situational needs [21].
- **Conclusion 3: - Customization of Agile GSD:** Eighty percent of the studies reported during the literature review that most of the agile practices such as sprint review and daily standing meeting are helpful in mitigating the core GSD challenges related to communication, coordination, and control but don't provide any mechanism for agile tailoring and customization as per organization, application and project situation. Different researchers have given their contributions in situational agile fragments, but still more improvement is needed in this critical area as evident from the literature. Bannerman et al. [22] describes about the success stories of agile GSD projects, but they are silent about the selection of appropriate agile method in GSD setting. Our investigation reveals that to support the use of SME principles, Agile GSD teams can be tailored and their practices customized as the situation demands.

To investigate those areas of research where still improvement is required and how we mitigate these identified gaps with supporting evidence from selected studies is shown in Table 2.2. We started our summary of the outcome of literature from the leftmost column, which consists of research problems that were identified from literature review. The literature review also helped us to find the existing solution related to our problem. Then we perform in depth analysis of existing solutions, however, a few gaps still exist [37,59,61,62,63,73,74,75,96,101,129, 130,131,132,135]. To overcome the existing gaps, we present our proposed solution.

Table 2.2: Summary of Outcomes of Literature

Research Problems	**Existing Solutions (Method and Techniques)**	**Still Gaps**	**Proposed Response**
Lack of team coordination	Video conference, Skype	Bandwidth Issue, delay	Generic Model to support team coordination, More teams work simultaneously
Knowledge Sharing	Discussion Board	No proper handling of repositories	Knowledgebase Repository
Domain Knowledge	No classification GSD terms	Didn't consider project characteristics for situational methods.	Provided *3C* Taxonomy to mitigate this issues.
An Appropriate agile method adaptation	A metamodel is used for decomposing the agile methods	Provided no mechanism for tailoring agile methods.	Same idea apply also Prepared agile fragments for method adoption
Problem in Agile customization and assessment	Agile methods customizing based on assessment and measurements	High involvement of method engineer for customization and assessment	Provide an approach for customization of Agile GSD
No predefined heuristics for popular agile methods tailoring	Methods for tailoring Extreme Programming	Methods only for one agile methodology tailoring e.g. XP	Methods defines for other agile methods such as scrum for tailoring.
Issues for agile GSD teams such as configurations and changes	Solve the configuration issues up to certain level	Provided no mechanism for agile method tailoring based on configuration issues	Provide workflow to accommodate *3C* challenges
Need a method for agile fragments selection	CAME tool for agile methodologies fragments selection	No proper interfaces for fragments definition and situation specification application	Formulate a prototype to overcome the situation specification
Lack of Systematic Agile GSD Methods	Shorter sprint Length	No proper Generic model	Provide *3C-SAME* Model
Lack of customization in order to select suitable method	Shorter Sprint Length Additional meeting	Cannot provide systematic Approach	Provide *3C-SAME* Framework

2.7 A Broader Overview of the Proposed Solution

In this section, a broader technical solution for identifying challenges is presented, established on the results of the literature review and empirical reports. Our broader solutions mainly focus on each identified challenge, collected from literature, past best practices from published cases and by defining a new one if not available in the previous two cases. Literature analysis describes many success stories about the trend of applying GSD with Agile practices in distributed environments. Hence, there is a dire need to explore the challenges that globalization entails, especially those concerning coordination and communication with diverse themes. However, it has also been discovered that every challenge has its diverse semantics, in line with diverse terms. Therefore, we proposed *3C*situational taxonomy (See Chapter 3) up to three levels to facilitate practitioners.

In fact, practitioners require more attention to fragile traditional agile methods which may be tailored according to the GSD environment by applying situational method engineering principles. Therefore, there arises a need to focus agile distributed teams and understand control, communication and coordination ratification. Hence, we argue that there is a dire need to explore such a framework not only to combine Agile practice with the GSD, but for an in depth understanding of SME. It should also be exhibited and adapted to distributed situations to mitigate those challenges in order to get the benefits of a flexible approach. Keeping in view the need of the current era, we proposed 3C-SAME (3C-Situational Agile Methods for Global Environment) Framework (See Chapter 4) to formulate the Situational Agile methods. Key components of our proposed approach are given below:

- Provide a situational taxonomy for better understanding of GSE terms and jargons for practitioners and researchers. The existing taxonomies [46] are up to level 2 and our proposed taxonomy is up to level 3.
- Our proposed approach consists of three types of activities: pre development activities, development activities and post development activities, whereas the other frameworks did not mention the pre and post activities and only focus on development activities [67].
- Our approach has a basic generic model, whereas other frameworks did not provide any generic model such as Emam. et al. framework [48]. In addition our approach is integrating two popular approaches such as Agile practices with the GSD, whereas the other similar attempt was only conceptual from Emam et al [48].
- Our approach provides additional feature of customization and adoption with Agile GSD practices whereas previous attempts such as Abad et al. [21] only provide customization limited to agile and not with Agile GSD.
- Our approach has adopted international product development ISO/IEC 24744 including SPEM meta model, which also used ISO and IBM for their development purpose, while others used simple meta model [68].

- Our proposed framework is also evaluated using working prototype, in order to extend and enhance the situations for Agile GSD teams. It has also provided a method composition according to organizational needs.

2.8 Chapter Summary

In this chapter, we provided the overview and objective of the literature review. In answering the Research Question (RQ1): 'What does the existing literature say about the Global software Engineering possible challenges and classify them in hierarchy?' we briefly present the systematic mapping review and its associated components. As the result of a literature review of existing research, we presented GSD challenges and their solutions. For the second research question (RQ2): 'What types of Agile solutions have been suitable for GSD and what was their impact on GSD Challenges' we have identified the Agile based solution in order to overcome the GSD challenges. At the end, we defined the outcome of the literature review and broader overview of the proposed solution.

Furthermore, we have also clarified that the behavior and active structure of the 3C-SAME framework focuses on optimal usage of coordination to handle situations during Agile GSD development.

Chapter 3: A Situational *3C* Taxonomy Creation

3.1 Chapter Overview

In the previous chapter, we presented a Systematic Mapping Review (SMR) of the relevant literature on and around the topic of Global Software Engineering (GSE) and Agile practices. We described GSE challenges with respect to distributed development environment. We also performed a critical analysis in GSE, focusing on the available taxonomies, techniques, approaches, and models.

The main objective of this chapter is to classify the issues, problems, challenges, threats and concerns of the distributed environment. The chapter also presents necessary investigation that helps in clarification, understanding and analysis of the identified issues of the dispersed environment. Special consideration has focused on issues, challenges, root causes of the challenges, situational factors and contextual factors that are of the highest significance and require immediate attention. Issues and their related element identification and clarification are mainly done through industrial survey, systematic mapping review and case studies.

We believe that, the identified GSE challenges must be carefully explored and studied to prepare the taxonomy of the challenges in 3C. This prepared term 3C has been found from the list of challenges that have been described in detail in chapter two. However,. GSE challenges have been presented in various studies in a disorganized manner and have not yet been classified. Also, we have been connected to the GSE industrial problems and challenges. Here 3C abbreviation has arrangement of Coordination, Control and Communication in the context of distributed development environment. The taxonomy comes from Greek word taxis signifying an arrangement or division of concepts. Another prospective of taxonomy is the science of classification as per specification or concurrence of the system. Furthermore, it is applicable with the resulting catalog to provide help to researchers and practitioners alike for discussion, analysis, or information retrieval. This must be done keeping in view the distributed environment so that the identified issues can be managed effectively to handle a particular situation.

Furthermore, the purpose of this section of study is to investigate the state of practice in GSE, as described in research objective 2, outlined in chapter 1. Furthermore, with regard to what is generally perceived but not proven about GSE in the environment, research Questions obtain answers highlighted in chapter one to formulate the taxonomy of 3C. The purpose of 3C taxonomy is to catalog or categorize the knowledge management concepts in a hierarchical manner [77]. It is, therefore, important to define types of hierarchical associations such as classes, subclasses, and super classes [78] e.g., "automobile" is the main class and its subclass is "motor vehicle". Similarly, super class of "automobile" is "motor vehicle". It is therefore important that proper understanding of important issues, problems and concerns are designed.

It is important to highlight the fact here that historically no similar explanatory research has been conducted for the GSE domain that helped in understanding, clarification and analysis of issues, challenges and situational factors of the distributed environment. Several challenges are identified in this chapter after industrial survey, published reports, published case studies, literature review and through study of the practice of ethnography guideline. Although all the identified challenges have been given due consideration, specific focus is laid on the issues of 3C with high priority. In addition to this, 3C has also been studied in detail in this chapter in order to develop its obvious understanding. It is also vital to note that this chapter exclusively focuses on soft issues of GSE environment such as team coordination breakdown and lack of communication. This chapter not only identifies challenges of GSE, but also helps in understanding the prioritized issues, problems and concerns.

The rest of this chapter is organized as follows: section 3.1 presents a detailed overview of this chapter, section 3.2 presents the analysis of existing Taxonomies, section 3.3 describes the description of the major 3C components, section 3.4 presents the process of creating 3C taxonomy, section 3.5 consists of 3C Situational taxonomy creation steps, sections 3.6 presents the taxonomy evaluation by the practitioners , section 3.7 presents the 3C taxonomy value addition, section 3.8 describes the discussion over practical implications and section 3.9 presents the summary of the chapter respectively. It is important to specify here that the literature review in this chapter has been done only to obtain the information regarding the challenges of GSE. Furthermore, we enhance and further improve the results of our review of theory by first investigating what has been confirmed about GSE practice in the available literature. At the end, we present a qualitative and quantitative survey of the taxonomy of GSE challenges consisting of a series of in-depth interviews with experts in GSE domain.

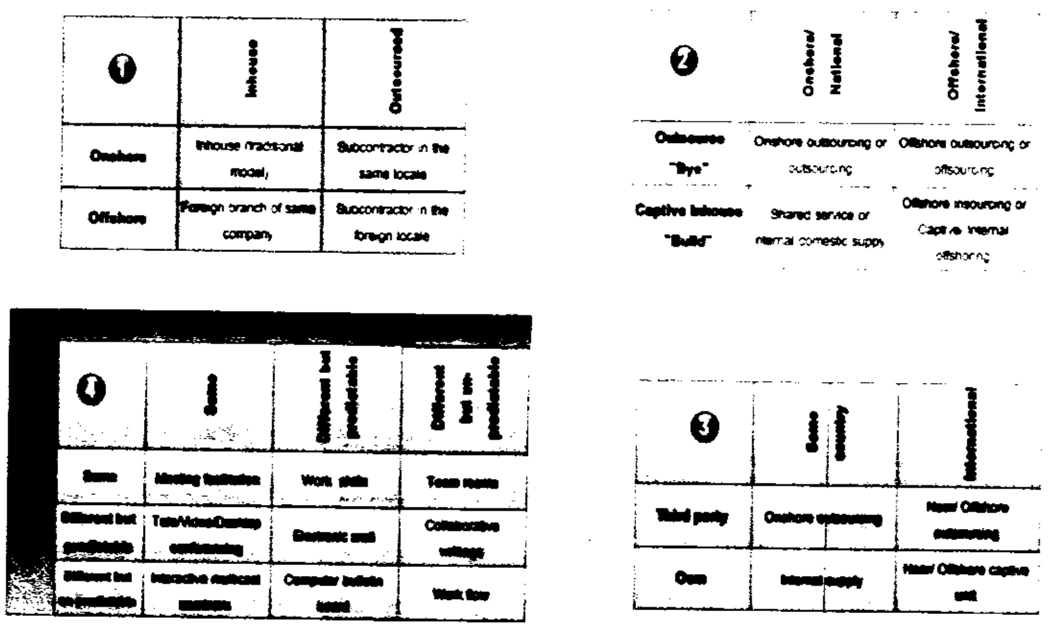
3.2 Analysis of Existing Taxonomies

In this section, we explicitly focus on the compilation, identification and classification of the challenges revealed in the existing literature. While extensive literature review in taxonomy based on empirical evidences is not available, it provides a comprehensive platform for the compilation of existing knowledge that practitioners can use to find evidence. However, some evidences of literature pointed out GSE as a general challenge existing in the distributed software industry. Therefore the design of 3C Taxonomy is fundamental to a systematic review of research contribution.

Taxonomy includes the jargon used for describing the relationships between terms. Taxonomy which is suitable for distributed environment is developed [83] in order to understand the communication and coordination terms between dispersed teams. Another taxonomy as shown in Figure 3.1 (Part-1) focuses on cooperative efforts, while the second part of the Figure 3.1 is useful for project development and their monitoring mechanism. The typical communication model for near shore and offshore has been presented in Figure 3.1 (Part-3). The challenge encountered by the GSE researcher here comprises a personal urge to do an extensive literature review to gain overview about the area under study. Gumm has developed one of the most associated efforts in the identical taxonomy of distribution direction [77]. However, the study is more descriptive in the distributed environment and builds on the causal dependencies between sharing threats [78].

There are a variety of taxonomies of sourcing strategies which have been found. Mostly they are established on the integration of dispersed dimensions and association structure among the distributed organizations [79,80] See Figure 3.1 (Part-4). In the previous studies classifications applied popular terms like offshore, onshore and outsource. Carmel et al. [85] explored the contemporary form of outsourcing which is being applied for GSD teams to overcome the distance challenge.

A few contemporary literature evidences also show the existence of knowledge clusters in the GSE domain[18]. Prikladnicki et al. [81] has characterized the major difference between the term offshore associated with internal off shoring outsourcing. Smite et al. [62,82] argue that there is a dire need of classified collaboration modes for inter-organizational and intra-organizational.



❶	Inhouse	Outsourced
Onshore	Inhouse fractional model,	Subcontractor in the same locale
Offshore	Foreign branch of same company	Subcontractor in the foreign locale

❷	Onshore/ National	Offshore/ International
Outsource "Bye"	Onshore outsourcing or outsourcing	Offshore outsourcing or offsourcing
Captive Inhouse "Build"	Shared service or internal domestic supply	Offshore insourcing or Captive internal offshoring

❹	Same	Different but predictable	Different but unpredictable
Same	Meeting facilitation	Work shift	Team rooms
Different but predictable	Tele/Video/Desktop conferencing	Electronic mail	Collaborative writing
Different but unpredictable	Interactive multicast seminars	Computer bulletin board	Work flow

❸	Same country	International
Third party	Onshore outsourcing	Near/ Offshore outsourcing
Own	Internal supply	Near/ Offshore captive unit

Figure 3.1: Existing Related Taxonomies

Another study, introduced an approach for query taxonomy generation which facilitates web users' search demand in a hierarchical fashion [78]. Paulo et al. [83] specified that there were no previous usability studies for the digital knowledge or for social repositories. Therefore they proposed visualization taxonomy which may effectively apply for large scale data analysis. With the emerging trend of taxonomies, Gorman, presented a shared taxonomy to provision enterprise knowledge portals [82].

As a result, there is a need to pay attention to establishing an empirically based taxonomy, which comprises a strong narrative to evaluate taxonomy through industrial experimentation.

GSE domain had a variety of challenges in general, but few common challenges that have been described in many studies from various angles. Paulo et al. [83] performed a structured literature review to identify the possible list of challenges, frameworks and models of distributed development. They investigated 54 different studies of the particular domain, out of a total of 266 for all 30 identified challenges declared as the main challenges. These five challenges were called major challenges and a commonly quoted categorization for challenges caused by globally distributed development is [85], illustrated in Figure 3.2 along with a short description.

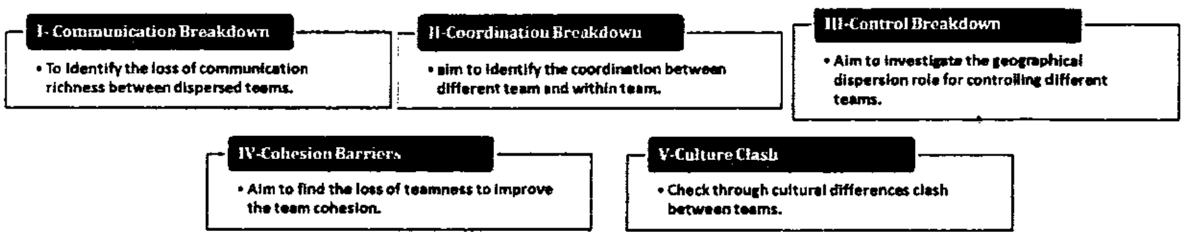

I- Communication Breakdown
• To Identify the loss of communication richness between dispersed teams.
II-Coordination Breakdown
• aim to identify the coordination between different team and within team.
III-Control Breakdown
• Aim to investigate the geographical dispersion role for controlling different teams.
IV-Cohesion Barriers
• Aim to find the loss of teamness to improve the team cohesion.
V-Culture Clash
• Check through cultural differences clash between teams.

Figure 3.2: Major GSE challenges

3.3 Description of the Major *3C* Components

The primary objective of this section is to introduce a theoretically comprehensive and practically suitable description of *3C* (*Coordination, Control, and Communication*). Furthermore, it also provides constructive and productive solutions to the challenges identified by integrating the theories and empirical studies of global software engineering, which provide a base for the construction of Taxonomy of *3C*. There are many challenges found in literature regarding GSE domain, but we choose the most common, based on empirical evidences, commonly known as software industrial evidences i.e. Coordination, Control and Communication.

I-Coordination

Coordination is activity, where people work together under the dispersed parts of the organization. It is also used to perceive the problem of sharing, integration and transferring knowledge. This section describes team coordination literature as systematized around the agile software development in a distributed environment. In 1988, Malone introduced a multi-disciplinary theory of coordination from the fields of organization theory and computer science [86]. Gregor et al., believed that taxonomy was a scheme based on discrete decision rules for categorizing phenomena [89]. The key idea in geographical distance is that multisite development also affects team coordination [16, 23]. Nowadays Computer organization need to use their existing resources as efficiently as possible and centralize their resources on a global scale from different sites. In order to use the centralized resources optimally, an efficient team is very necessary.

Naizi et al. [40] highlighted reasons for project failure, such as lack of team coordination and communication. Rodríguez et al. [38] focused on coordination tools to support global teams. As a result, the software manager faced a different problem to manage and coordinate the global teams in varying situations.

Table 3.1: Fundamental Description of Term Coordination

Definition	**Research Field**
"The Integration or linking together of different parts of an organization to accomplish a collective set of tasks" [90].	Organisation definition
"People work collectively; The work is interdependent; A goal a task or piece of work is achieved" [91].	Organisation Studies
"Coordination is the managing of dependencies between activities" [88].	Interdisciplinary
"Coordination means the spatial and temporal synchronization of overt behaviours of two or more people so that those actions fit together into an intended spatial and temporal pattern" [92].	Teamwork studies
"Coordination is then perceived as a problem of sharing, integration creating, transforming and transferring knowledge"[93].	Knowledge Management

Nuevo et al., [47] illustrates that the team coordination and customization issues required serious attention in multi-user environments. But despite the obvious need of an appropriate method to overcome the geographical distance, face-to-face communication is a complex task, and tailoring and customization is still a gap between theory and practice. Meanwhile the Quershi et al. [54] proposed reduction of coordination costs in Global Software Development. In particular, activity may help to reduce the coordination cost to mitigate the effect of socio-cultural and geographical distance.

Based on available literature we are proposing a taxonomy for 3C situational taxonomy in Figure 3.6 for distributed environments.

II-Control

Another very important challenge in distributed development was to control the dispersed teams, which exist at every stage of global software development. The control acts as the macro activity, applied in the overall project management. It is used to control the team budget, schedule, and the quality of the project. Team Size is used as a parameter and confirmed by the team leaders. The size of a project team plays a vital role in the quality of a team's collaboration and its effectiveness is the key to the project success [94, 84]. As the team size increased it becomes more difficult to interact with each other and this creates a lot of problems when working in dispersed environment [93].

Another vital challenge faced by teams working in dispersed environments is that quality and team efficiency may be affected between the team controls. Teamwork is directly associated with the efficiency of critical projects. The relationship between teamwork and quality plays an important role in team management. Internal control is a procedure, affected by the team management and other personnel, designed to provide quality assurance regarding the accomplishment of objectives relating to daily decision making.

The outcomes and the emergent properties achieved by Control are as under:

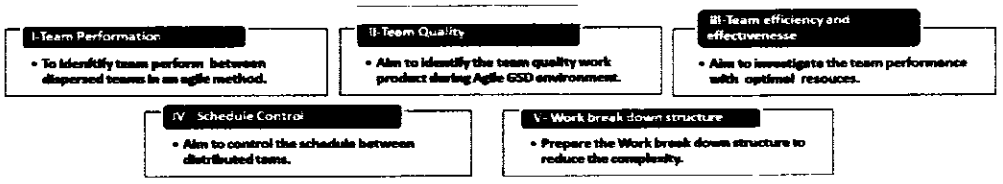

I-Team Performation
• To identify team perform between dispersed teams in an agile method.
II-Team Quality
• Aim to identify the team quality work product during Agile GSD environment.
III-Team efficiency and effectivenesse
• Aim to investigate the team performance with optimal resouces.
IV Schedule Control
• Aim to control the schedule between distributed tams.
V- Work break down structure
• Prepare the Work break down structure to reduce the complexity.

Figure 3.3: Control Outcomes

III-Communication

Communication is another essential challenge that arises due to varying time zones, dispersed teams and cultural variation issues. When two teams were situated at dispersed environment, they face additional difficulty for conceptualization and to understand the viewpoint of each other. Many developers working in dispersed environment do not have english as their native language. Keeping in view the politeness factor during a conversation between dispersed team communications, issues can be minimized [60].To handle this challenge, direct communication and instant messaging can be made use of instead of the email. To avoid confusion, mediators can play a vital role to solve communication challenges due to language and cultural differences among distributed teams [85]. Handover and maintenance teams in a distribued environment face a number of queries about system maintenance and require a dedicated offshore handover person to handle these types of problems [84]. Shared activities between on-site/offshore development environments include business conception, detailed design and testing that are used to improve team spirit [97].

3.4 Process of Creating *3C* Taxonomy

This study selected a rigorous research method known as Grounded Theory [30] for the creation of *3C* GSE taxonomy as presented in Figure 3.4.

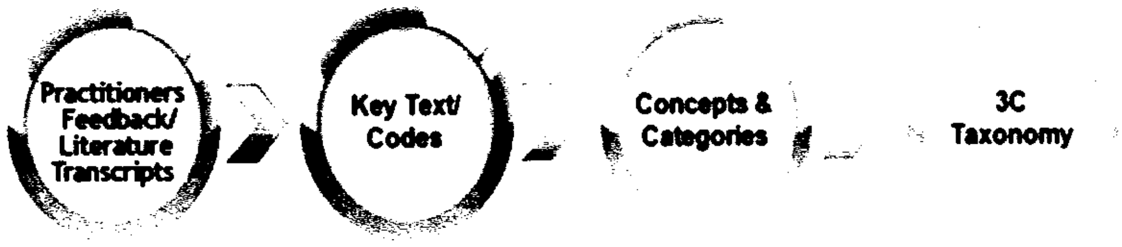


Figure 3.4: Process of Creating 3C Situational Taxonomy

3.4.1 Data Collection

We conducted data collection through face to face, semi structured interviews with GSD practitioners applying open ended questions in the taxonomy creation. In doing so, we considered three aspects. In the first and foremost step, the domain terms and scope of study was identified through the critical review of literature transcripts [94]. Several online databases were considered under the domain of distributed software development for the scrutiny of the literature. GSD experts were interviewed for the better understanding of GSD terms commonly applied in distributed software industry. Furthermore, criterion for the evaluation of the GSD terms through real world case studies was helpful in finding empirical data from industry.

The context of the study is to evaluate the terms of taxonomy. Pilot study was designed followed by the context analysis. We have selected a computer based organization in order to execute the pilot study. The organization has over fifteen years of GSD experience. We selected a total of twenty five (25) participants, randomly from the organization. Among the total participants. In addition, the participants in the group followed the routine development using Agile GSD practice along with their previous knowledge. We have conducted fifteen (15) interview in couple of GSD based organizations.

3.4.2 Data Analysis

We applied open coding to investigate the interview transcripts into key text. During the data collection, a comprehensive list of GSD challenges was identified [34]. We drafted terms on the basis of the collected definitions; called key text which is further analyzed using multiple regression models. As this research is relevant to exploratory research, it was essential to investigate the multiple regression impact. We perform this impact analysis for identifying critical challenges of GSD and their impact on Global teams working in dispersed environment. Numerous emerging concepts were introduced by applying "constant comparison method".

Heath et al. [95] describes about the open coding technique which is applied in preliminary phase of the theoretical analysis for discovery of initial categories, concepts and their properties. Furthermore, it is a procedure that refined the data assembled associated with concepts and categories. Response of the interview along with observations composed of literature was investigated through key facts. An illustration of key point analysis is given as under:

INTERVIEW QUOTATION: "*Although we can minimize the effect of team coordination but we can't eliminate the problem completely due to geographical distance barrier*".

KP1: Effect of team coordination directly associated with project failure.

Code: Effect of geophysical distance.

Figure 3.5 illustrated about the codes that were extracted from the interview. Furthermore, to evaluate the codes, they were compared to each other in order to examine the relevance. Figure 3.4 illustrated the conceptual view of grounded theory while Figure 3.5 indicates how related codes with the shared theme were assembled to produce classification, which was an advanced level concept making relationships clear between codes and concepts.

3.4.3 Generating a Theory

The concluding stage of grounded theory is producing a theory which is also recognized as "Theoretical Coding" (TC). It comprises of conceptualizing how the classifications and their associated properties narrate to each other as a hypothesis to be formulated into a theory. (GT) is more effective when we proposed a theory [97,98]. We applied the grounded theory in order to formulate the *3C* Taxonomy. On the basis of progressive grounded theory growth, taxonomy was proposed. All components of this classification of GSD challenges have been integrated in *3C* taxonomy.

Similarly, by applying recognized codes and classifications from conversations and literature review open coding was applied to design the grounded theory.

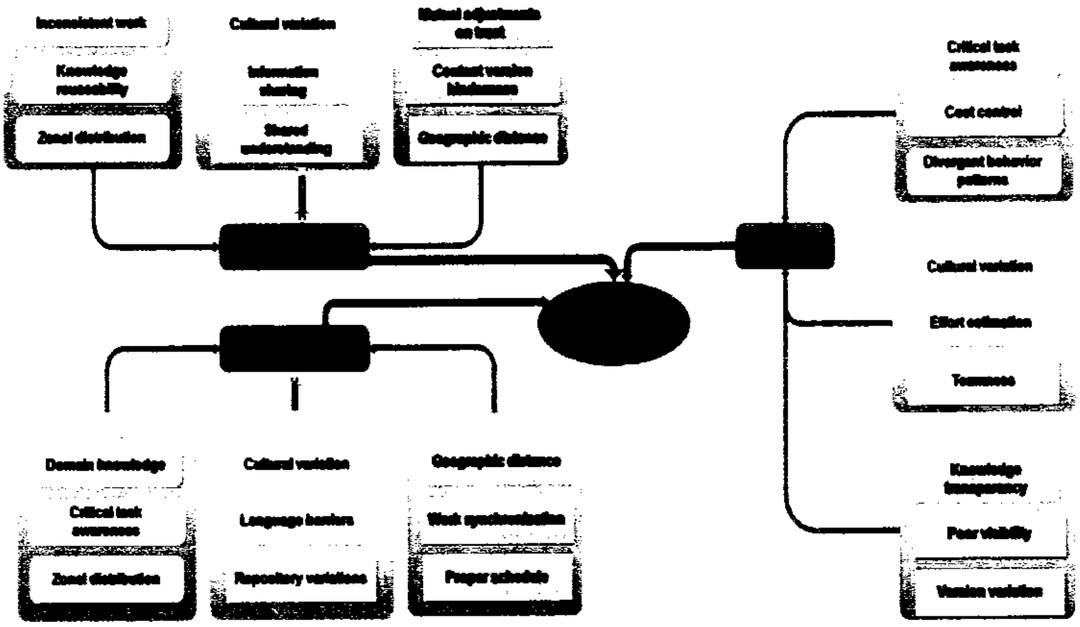


Figure 3.5: Emergence of *3C* Theory from Concepts

3.5 3C Situational Taxonomy Creation

The taxonomy may be generated by illustrating the associations from simplified theories to knowledgeable terms, which supports joining expressions in a particular domain or for a specialized subject. Therefore, taxonomy may be defined as a classified catalog of a subject, issue or domain. Formerly, taxonomies derived from illustrating animal species are formally called class and their associations are called properties of class.

The purpose of creating a taxonomy in GSE domain would be to identify the similarities and difference between the distributed development challenges, through the analysis of the challenges faced by GSD teams led by several interesting conclusions and dimensions. The analysis suggests the following three main dimensions: I-Coordination, II-Communication, III-Control, for the explicit understanding of GSE context challenges. On the other hand Agile practices are used for development approach. We present Agile framework in the subsequent chapter 4, as development approach for global teams. Currently, GSE challenges and their root causes, situational factors, and their contextual factors are available in a disorganized manner and therefore, there is a dire need to classify them into hierarchical manner to facilitate practitioners and researchers. On the hand *3C* taxonomy provides the fundamental challenges of GSE, along with derived challenges, consequent causes and their situational factors and contextual factors in an organized manner [69, 71, 72, 106 and 127]. The proposed taxonomy would potentially result in improvement of practitioners understanding. The optimal usage of *3C* Taxonomy is to synthesize the domain knowledge.

The *3C* Taxonomy (See Figure 3.6) classifies the challenges identified through literature review and expert survey. It is further divided into five levels as presented in Figure 3.6.

- The starting point is GSE Challenges, i.e. the main cause behind the basic circumstances is global distance in time and space between global teams, which makes it impossible to communicate and coordinate. They are directly linked with difficult GSE settings as well as provide a basis for further challenges, such as Geographical (informally known as space) challenge and Temporal (informally known as Time) challenge. Root causes are the basic reasons behind the issues in GSE that can be tackled using optimal solutions. The major root causes are discussed in three situational factors: *Application Domain, Project Organization and Organization Developing Environment* in order to accomplish an ample perceptive of the contextual factors. Moreover, the root causes are interlinked and three situational factors have been identified from different sources of literature and empirical studies.
- The next level is concerned with Derived challenges, which are the conditions that may not be found in every distributed development environment and their effects on project can often be avoided with the proper working approach like a time zone mismatch and difficulty in face to face meeting. The derivative causes include lack of

communication, lack of control and coordination breakdown between the GSE teams and improper usages of resources. Different backgrounds also involve differences in implicit knowledge causing confusions and wrong predictions.

- After the Derived challenges, upcoming level is Consequent causes that are not forever part of GSE. However, they occurred due to improper handling of derived challenges. First consequent challenge is clarity in requirement. Incomplete requirement can cause project failure. Shared understanding is another vital challenge for GSD project.

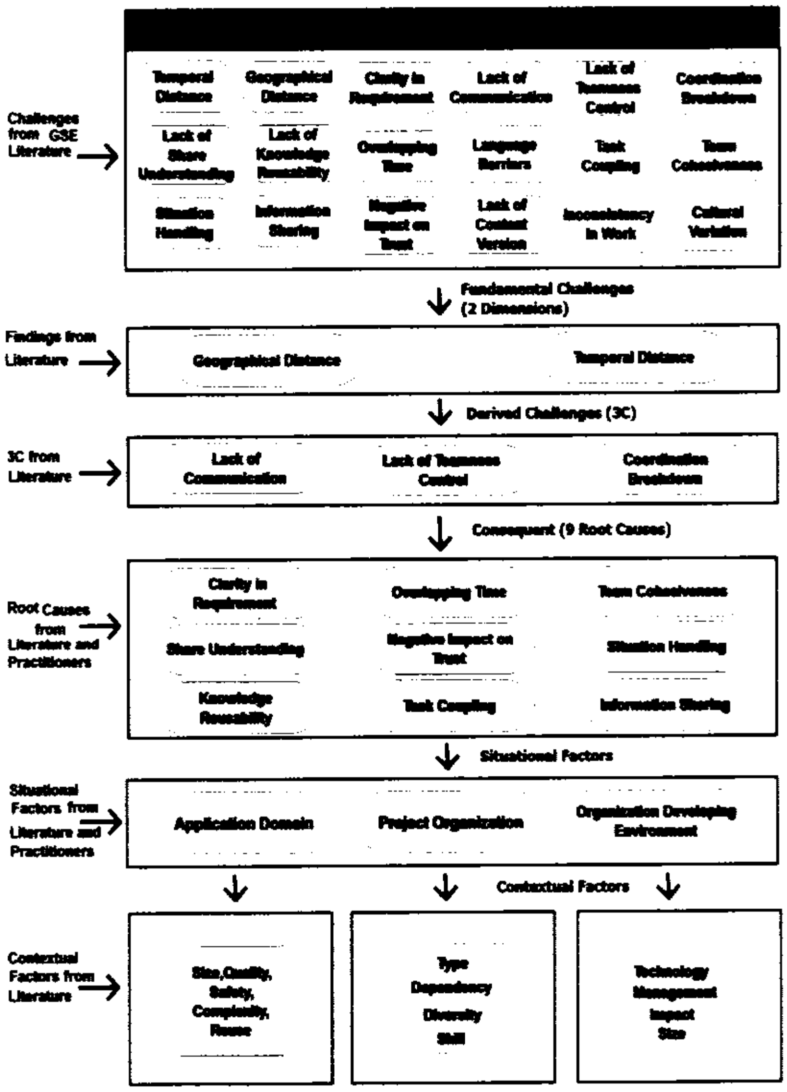


Figure 3.6: The Proposed 3C Situational Taxonomy for GSE Challenges

- Furthermore, knowledge sharing is also used for GSD teams to reduce the cost, time and improve the quality. The role of trust has been always critical in cooperation as it is need to make ideal contracts, covering all aspects of a relationship, and can be very problematic if it doesn't exist. During the GSD working environment, teams are scattered and due to time zone difference, it is much more difficult to organize meetings in overlapping time. Another vital challenge is information flow, which can cause delay among GSD teams. Task dependency such as the outcome module A may have the input of module B, which is also a vital issue in GSD setting. The third derived challenge is coordination, which is also caused by lack of cohesion in software module. Lack of information sharing has also created problems for GSD managers.
- In order to overcome the complexity of computer systems, project situations are distinguished by applying a set of features formally known as situational factors. It implies few contextual factors such as complexity, size, staff experience, diversity skill and complexity. In the distributed situation, every challenge implies serious attention to minimize the project failure rate and that can affect cost overruns, schedule overruns and ultimately project failure and hence project quality suffers.

3.6 Taxonomy Evaluation by the Practitioners

To make our taxonomy clear and concise, one term for each GSE concept was agreed upon. Different options for each term were given to the experts, firstly on preferences basis, i.e., by selecting terms from a set of synonyms, and then on non-preference, giving preference for disapproving the proposed terms or signifying new terms.

Roles and responsibilities were assigned to the participants as per their organizational hierarchy. The project manager maintains the profiles of every individual participant. Allocated tasks, responses made against the allocated tasks, interaction time with proposed work and situational contents for participants were a few items of the participant profile. Keeping in view the experimental context, all participants had at least sixteen years of education in computer science and had basic knowledge of GSD terms. The time line for the experiment was twenty five days. The participants were cooperative due to the involvement of their top level management. A project manager remained within experimental place in order to motivate and provide support if required, such as reading the question statement and understanding the Agile GSD terms. Furthermore, hypothesis evaluation was conducted in order to investigate the cause and its possible effect on the relationship between the samples that have been used for the experimental study.

The decisions were made on the basis of following arguments:

Approved terms (expert agreement or consistency between different terms)

Suggested terms for non-preference.

Rejected terms

Disagreement among experts

Inconsistency of terms

The identified definitions were communicated by investigators on the basis of literature. Experts could be responsible for accepting or rejecting or proposing his/her own definition. Final accepted improvements were in the final version of the taxonomy.

In order to prove our theory we proposed ten null hypothesis. Each hypothesis represent categories defined for our proposed 3C taxonomy.

H_0^A : Geographical distance does effect on rate of coordination and communication.

H_0^B : Trust is directly affected due to mutual hindrance in a distributed environment.

H_0^C : Lack in content version is observed due to geographic distance.

H_0^D : Lack of knowledge reusability due to temporal distribution effect coordination, communication and control.

H_0^E : Inconsistency in work is also observed in coordination.

H_0^F : There is certain level of cultural variation that effects communication, coordination and control in distributed environment.

H_0^G : Lack of domain knowledge hinders effective communication in distributed environment.

H_0^H : Immature task coupling results in less synchronous work.

H_0^I : There is difference in the level of communication due to situation handling.

H_0^J : There is a certain level of difference while controlling teams in distributed environment.

Table 3.2: Questions from the questionnaire representing hypothesis $H_0^A - H_0^J$

Variable	Question	Rating Scale
A	Rate the level of difference in communication due to increase in geographic distance.	1…5
B	Does trust matters for mutual hindrance for communication in distributed environment?	1…5
C	Does communicating content version difficult due to geographic distance?	1…5
D	Rate the difference level of knowledge reusability due to temporal distance?	1---5
E	Does lack in coordination results in inconsistent work?	1---5
F	Does cultural variation effect communication, coordination and control in distributed environment?	1---5
G	Does lack of domain knowledge hinders effective communication in distributed environment?	1---5
H	Does Immature task coupling results in less synchronous work?	1---5
I	Does situation handling effect communication due to geographic distance?	1---5
J	Rate level of difference while controlling teams in distributed environment?	1---5

In order to get interview responses from domain experts, following questions were formalized as shown in Table 3.2. A variable representing each hypothesis and associated question is shown in Table 3.2. The rating scale is also defined for the questions. Each question represents the null hypothesis which helps in getting responses for both the traditional software development and distributed agile software development.

Responses extracted on the basis of questionnaires were statistically analyzed in order to prove the proposed hypothesis.

Table 3.3: Quantitative analysis of responses for hypothesis. $H_0^A - H_0^J$

[illegible]			[illegible]	[illegible]	[illegible]	[illegible]	[illegible]		[illegible]
***H_0^A*:Communication variation**	DA	5	3	2/4	3	6.3	0.89	8.5	0.46
	Tra	5	2.4	1/4	2	4.7	1.01		
***H_0^B*: Trust and mutual hindrance**	DA	5	2.4	1/5	2	5.6	1.49	12	1.0
	Tra	5	2.2	1/4	2	5.4	1.46		
***H_0^C*: Content version**	DA	5	2.8	3/4	4	6.3	1.32	8.5	0.45
	Tra	5	2	1/3	2	4.7	0.89		
***H_0^D*: Knowledge reusability**	DA	5	3.8	2/5	3.5	7.4	0.4	5.5	0.16
	Tra	5	2.4	1/5	2	6.4	1.49		
***H_0^E*: Inconsistent work**	DA	5	4.4	4/5	4	7.2	1.75	1.5	0.01
	Tra	5	2.8	2/4	3	6.7	0.74		
***H_0^F*: Cultural Variation**	DA	5	3.2	2/5	3	8.1	1.16	2	0.02
	Tra	5	1.4	1/2	1	5.2	0.48		
***H_0^G*: Domain knowledge**	DA	5	1.4	1/2	1	3.4	0.48	2	0.06
	Tra	5	2.6	2/3	3	7.6	0.48		
***H_0^H*: Work synchronization**	DA	5	2.8	1/4	3	7.2	1.16	7	0.29
	Tra	5	3.8	2/5	4	6.7	1.16		
***H_0^I*: Situation Handeling**	DA	5	4.2	3/5	4	6.6	0.74	7	0.28
	Tra	5	3.4	2/5	3	4.4	1.01		
***H_0^J*: Teamness**	DA	5	2.4	1/3	3	5.6	0.8	12	0.99
	Tra	5	2.4	1/4	2	5.4	1.01		

Statistical analysis was performed with the help of questionnaires, which helped in conducting the U-test and probability of each variable. In Table 3.3, DA represents Distributed agile while Tara represents the traditional approach. A total of 25 participants participated in the interview process and gave their responses as per defined rating scale. For conducting statistical analysis, mean, min/max, median, and mean rank were calculated.

Afterwards standard deviation was calculated and the U-test was also performed. At the end probability was checked on the basis of significant value. Significance value of 0.03 was set for comparing the probability. Cultural variables and instant work were not accepted in the null hypothesis as their probability was less than the significance value. Therefore, alternate hypotheses were selected for both the variables. The probability was checked as two folds.

3.7 *3C* Taxonomy Value Addition

The previous section validated the *3C* Taxonomy from GSE practitioners. The empirical result of this *3C* taxonomy work is value addition, which is helpful for distributed teams in order to improve their performance for distributed agile development environment. The results, including industrially published empirical results by other researchers were carried out. Based on the proposed *3C* Taxonomy, these results of the main challenges of GSE were evaluated. Moreover, the practitioner feedback to validate the taxonomy reflects the practical implications of the taxonomy, as perceived and evaluated by the experts.

This section of the taxonomy can be estimated as a diverse case as shown in Figure 3.7, of comprehensive case study research method where data is composed from empirical cases of agile GSE companies, that supports a cross case investigation when compared to GSE taxonomy[18]. Such a type of empirical study is where the researcher behaves as an investigator rather than contributor in the case study method [99]. Yin proposed an appropriate research method to evaluate the results. Its objective is to examine our contribution in real-life environment [18]. The case study research method is appropriate in natural setting when control over behavior is evaluated.

Context Analysis is the process to find out situational factors which collectively define the situation of a particular environment for which *3C* Taxonomy is to be evaluated. Case context is one of the critical stages as shown in Figure 3.8, because case selection plays a vital role sequentially to make the research context easier and improve the investigation procedure. It can be prepared, for instance, by applying particular situation of GSE challenges (i.e. Coordination, Control, Communication) when choosing domains for the case study GSE environment using agile methods exploration method [17]. At the start of this study we were also concerned to start assuming.

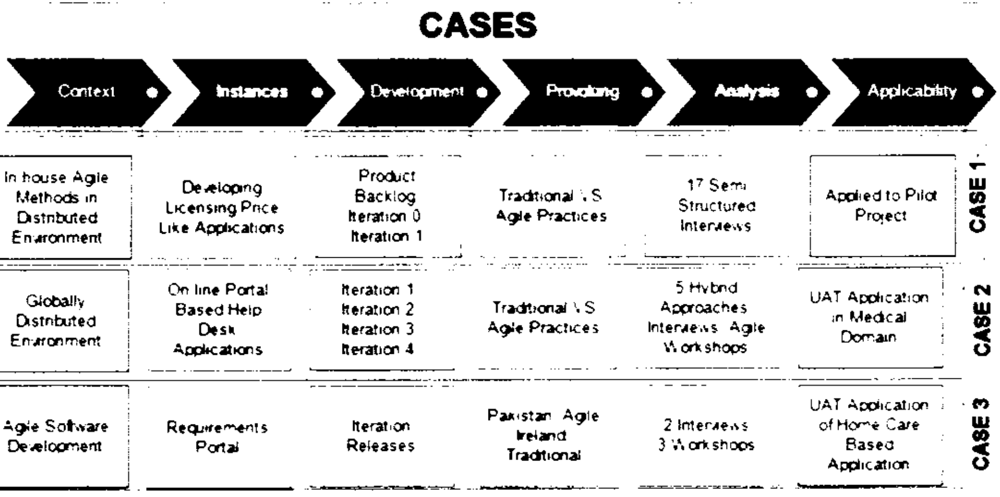

CASES
Context
Instances
Development
Provoking
Analysis
Applicability
In house Agile Methods in Distributed Environment
Developing Licensing Price Like Applications
Product Backlog Iteration 0 Iteration 1
Traditional VS Agile Practices
17 Semi Structured Interviews
Applied to Pilot Project
CASE 1
Globally Distributed Environment
On line Portal Based Help Desk Applications
Iteration 1 Iteration 2 Iteration 3 Iteration 4
Traditional VS Agile Practices
5 Hybrid Approaches Interviews Agile Workshops
UAT Application in Medical Domain
CASE 2
Agile Software Development
Requirements Portal
Iteration Releases
Pakistan Agile Ireland Traditional
2 Interviews 3 Workshops
UAT Application of Home Care Based Application
CASE 3

Figure 3.7: Three Cases Summary

The cases were selected for their different challenges, to provide a sound foundation for evaluation of the success issues of distributed environment.

The effectiveness of 3C taxonomy is composed on the jargons for making linkage in a distributed development team. The taxonomy is branched or branches off the particular term. We have identified the challenge areas like collaboration structure, knowledge management, etc. However, we are framing the taxonomy for contextual factors like size, quality, skill, impact size etc. The taxonomy is developed in response to the questions whose aim identifies the questions highlighted at the end of each challenge like which coordination mechanisms are suitable across sites. However, we have evaluated our proposed taxonomy, further by formulating cases based on the empirical study in real world software industry and that provides a proof of concepts.

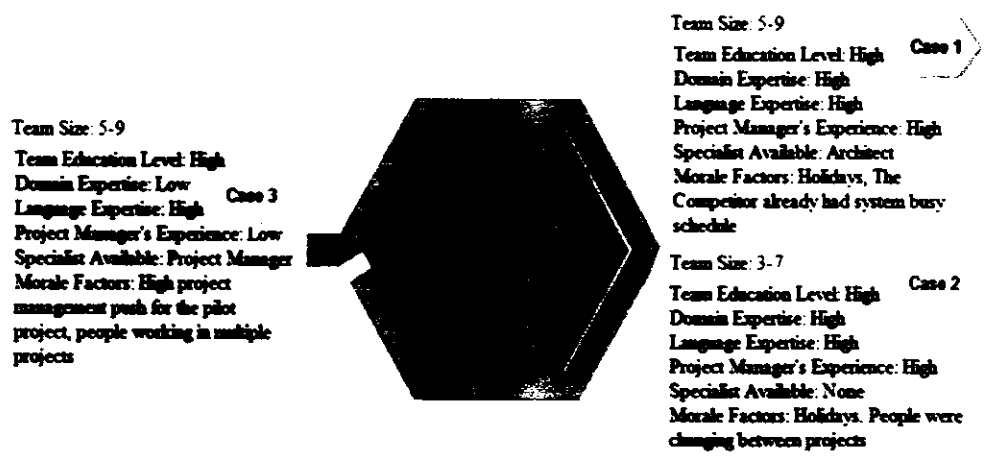

Team Size: 5-9
Team Education Level: High
Domain Expertise: High
Language Expertise: High
Project Manager's Experience: High
Specialist Available: Architect
Morale Factors: Holidays, The Competitor already had system busy schedule
Case 1
Team Size: 3-7
Team Education Level: High
Domain Expertise: High
Language Expertise: High
Project Manager's Experience: High
Specialist Available: None
Morale Factors: Holidays. People were changing between projects
Case 2
Team Size: 5-9
Team Education Level: High
Domain Expertise: Low
Language Expertise: High
Project Manager's Experience: Low
Specialist Available: Project Manager
Morale Factors: High project management push for the pilot project, people working in multiple projects
Case 3

Figure 3.8: Case Context Factors

Our proposed taxonomy has been designed to guide the researcher or practitioner in a hierarchical structure and also to provide the domain knowledge of GSE terms. In addition, we presented *3C* situational taxonomy in order to clarify the basic domain terms and domain knowledge before the commencement of the project which is shown in the Figure 3.7.This taxonomy is also useful to identify the branch of knowledge, formulating in particular variance of time and distance between teams using agile methods. The participant feedback was satisfactory about the taxonomy.

3.8 Discussion over Practical Implications

In response to potential challenges of GSE, we have established a diversity of expressions to illustrate variant communication, control and coordination modes in distributed projects, and diverse expressions are established to illustrate the related ideas. The major variance in the state of art terms used as different jargons related to the GSE domain was generalization-specialization relationships among the GSE terms.

Furthermore, inconsistency and tendency terms mutually exclusive meanings were refined and shifted into proposed taxonomy. When we manufactured results from different studies of distributed development and state of the art agile practices, it creates a lot of difficulty to make consistency and clarity in GSE terms along with coherence.

Our contribution emphasized on the different levels in the first and foremost level the major challenges like communication, coordination, control and these challenges were further subdivided into fundamental challenges and consequently these challenges were shifted into derived challenges. Then in the next level, we arranged consequent causes, situational factors and their contextual factors. We were, then applying empirical reports, extensive literature reviews and expert judgments to refine the terms and reached a final agreement on a common terminology. Furthermore, a few terms selected based on weak substantial evidences and few key concepts were chosen based on varying opinions of synonyms.

A few experts have shown serious, concern about the complexity and understandability of issues reported in the GSE taxonomy review. Our main intention of the proposed taxonomy was to make it possible for the researchers to understand and use the knowledge from research of the GSE domain into the agile practices. The proposed taxonomy contribution is provided as a guideline for agile teams in characteristic different situations and identifying related GSE terms.

The contribution of this study may be viewed in a different perspective. First and foremost, the study provided a situational taxonomy for practitioners. Secondly, it elaborated upon how classification of GSE terms can provide a base for empirical study and also compared it with the existing and related studies. In addition to this our proposed taxonomy was also evaluated with the help of a number of diverse real world scenarios. We believe that new area of research will be introduced and more depth case studies to map empirical cases to the proposed taxonomy with state of the art agile practices will be used for evaluation. Our observations about the agile practitioners were to avoid all possible misleading uses of similar terms.

3.9 Summary of the Chapter

Our objective is to establish a variable in terms to clarify a diverse meaning which is based on similar concepts in order to fulfill the research question, *RQ1. What does the existing literature say about the Global software engineering possible challenges and their possible classification?* There are many GSE terms that have been used in a distributed development environment, but these cannot be easily identified from glossaries. However, different authors are used conflictingly different terms and novice readers are often confused with exclusive implications. This leads towards a complex situation when practitioners try to draw conclusions on the productive results from different research studies. We conducted a survey to evaluate our GSE terminology for an expert's judgment on a common term. We accomplished that researchers are facing difficulty to select appropriate terms without a defined terminology for the reader's context.

The main emphasis of the proposed *3C* taxonomy is to facilitate the understanding and the ability to handover the valuable theory from research into Agile GSD practice. Our research contribution is to facilitate the Agile GSE practitioners in twofold. First and foremost, our target audiences were those researchers who will publish their further work and need a roadmap. Secondly, we address Agile GSE practitioners, who are fascinated in published empirical cases to evaluate their proposed work and furthermore, to provide a domain knowledge of GSE terms for practitioners. We highlight the significance of circumstantial knowledge in an empirical study, which must to be comprehensive. We try to overcome an inconsistent use of terms as an impediment, situation factors and contextual factors respectively.

The future research direction of our proposed study may be observed in two aspects. First, we proposed a taxonomy which provides a comprehensive guideline for new GSE researchers. Secondly aspects of the study show how the proposed taxonomy can be applied in other areas of software Engineering. Thirdly aspects of the study are recognized for identifying situational factors and also contextual factors implied to modern development to record observed evidences to evaluate our proposed taxonomy.

Chapter 4: A Situation Specific Framework Based Approach

4.1 Chapter Overview

In the previous chapter, we have discussed the challenges, opportunities and threats found in GSD domain. Based on the classification and comparison, we proposed a situational taxonomy and scope was limited to only three main aspects: communication, coordination and control within the context of situational distributed environment. On the other hand, there are certain situations in global software development which create difficulty to grasp the context and scenarios for GSD managers, practitioners and researchers, at the lowest level of granularity. Keeping this view, the proposed 3c taxonomy was designed to identify the possible challenges and implies contextual factors of the distributed environment. More obviously, GSD practitioners and researchers have faced difficulties in probing relevant research terms since this field has variety of keywords with ambiguous usage. The proposed 3c taxonomy enables such practitioners to classify and compare issues and their possible causes along with situational factors of dispersed environment.

Nowadays software industry trends have been shifted into several interlinked systems, including social (e.g., service system), technical (e.g. software) and socio-technical (e.g., information system) systems. The development of each system must be based on a systematic approach of the organization and its working style. Furthermore, systematic approach assures on time, within budget and quality product to the client. Therefore, in order to deliver high quality product quicker, as per client expectations, the role of agile been impelled [100]. We argue further, there is a need for a systematic approach to tailor agile principles for the GSD environment keeping in mind that agile principles may be helpful to mitigate the GSD challenges [101].

In this study, we propose a situational Agile framework incorporating GSD challenges by introducing situational factors for agile development activities. It is designed for multidisciplinary support on the basis of situational factors. The aim of this chapter is to further strengthen the argument that GSD managers and practitioners are unable to assess the appropriate contextual factors prior to commencement of any Agile project in a distributed environment. Emam et al.[14] highlighted that there is a dire need to understand what Agile practices are more appropriate for the particular situation of the project. We propose a 3C-SAME (3C_Situational Agile Methods Engineering for Global Environment) Framework, which is a conceptual framework and verified by using ISO 9126 standard. The scope of this chapter was restricted to only well-known agile and GSE theories, philosophies and best practices. It is also important to note here that analysis of theories and philosophies has been conducted purely by the use of real world case studies to evaluate 3C-SAME. The investigation of phenomena from state of the art software industry also involves constant feedback from appropriate participants.

According to the best of our knowledge and literature surveys conducted, there is very little attention has been laid for situational factors in agile distributed software development. Furthermore, there is no comprehensive research related to exploration, explanation and analysis that would formally help in customization of the agile methods for distributed environment even though various globally accepted threats require more concentration for further insight, at the lowest level of granularity.

We also highlight the significance and applicability of the proposed framework 3C-SAME in distributed agile development environment. Communication and coordination between the different kinds of team members and understanding both technical (requirement to hander over) aspects and social aspects (incl. Cooperative and Communicative aspects of human activities) of processes as well as understanding of the situational factors and conditions which affect how the organization can operate in a distributed environment is of great importance.

4.2 Background and Motivation for the Framework

This section presents the needs for proposing a situational framework by exploring contextual situations for which it is suitable. There are four aspects about software development described in this section. First aspect is that GSD manager and practitioners shift their development approach from traditional to agile; secondly; they prefer their development in a distributed environment; thirdly is an integration of both agile and GSD approaches and fourthly the customization of Agile GSD approach that utilizes the SME principles.

For the last four decades software development methodologies have been introduced for varied organizations and practitioners facing problems regarding the development [100]. The IT software industry possesses strict schedule and restricted budget in a working environment. Software development is becoming highly unpredictable due to creepy requirements, business scope, market trends and needs. Due to the shifting trends organization is shifting from traditional software development towards agile methodologies as they are light-weight and it has potential to fulfill the customer's desires and they are more rapid and efficient than the traditional methods. The light-weight agile manifesto was introduced in the 1990's called as agile practices [50, 133]. This newly developed method aimed at flexibility, rapid and effective development under changing and creeping situations, time constraints and uncertain conditions [103]. The state of the art analysis on the software development techniques and notations reveals some common pitfalls and problems related to the usage of agile methods [47].

On the contrary, GSD is an emergent area due to its fruitful output approach in which stakeholders are dispersed at different sites with temporal and geographical distance involved [105]. The agile practices, have also received significant criticism when they are used in distributed environment, because GSE does not support the concept of physical dispersion such as offshore development [12]. Furthermore, agile teams work on direct communication and continuous feedback, therefore, face to face communication, and stakeholder involvement is mandatory [47].

It is obvious that one single software development method cannot cover all the aspects. Furthermore, different success factors and viewpoints need to be considered in an approach which supports best practices of distributed environment as well as the integration of different Agile approaches is a significant consideration. Integration of Agile methodologies with GSE is gaining a greater edge due to its progressive trends, over collocated organizations in order to achieve better software product by incorporating the benefits of both. Distributed agile development itself has attracted IT industry due to its imminent dominance, such as reduced time to market, reduced cost and managing creeping requirements. The attention is required to integrate ideas from multipurpose framework which supports effective coordination activities in agile distributed development to support situation specific features.

Numerous research problems in the practice of the software development can be embedded in the integration of agile with GSD. Special consideration may be paid to contextual factors of agile software development in a distributed environment for the customization of Agile practices in GSD environment. Software development challenges vary from situation to situation and are also characterized on the basis of contextual factors such as project size, team size and requirement stability. It is generally recognized that it is not possible to have a universal method without any modifications used in all situations. The Situational Method Engineering (SME) was introduced as a remedy for the appropriate handling of situations in GSD environment. Hence SME aims at providing method construction techniques by reusing existing practices.

Furthermore, little re-use of frameworks, a lack of requirement change management and poor knowledge management are listed as recognized causes in software development methodologies in general and distributed environment in particular. This framework is expected to provide support to the GSD managers and practitioners in deciding where GSD and where Agile strategies are effective.

4.3 Framework Designing Process

Action research comes under the umbrella of qualitative research approach which is applied for the formulation of a situational Agile framework for GSD. Its objective is to improve practice through active collaboration between participants and researchers.

We select an action research method to formulate our proposed 3C-SAME framework. We follow the strategy of Canonical Action Research (CAR) which is the specialized form of action research. This provides a comprehensive set of steps with a focus on planning and conducting the research along with evaluation and reflection of research results [107]. It is vital to mention that CAR is one of the forms of Action Research Methodology that aims at solving an immediate (research) problem or alternatively as part of a community of practice specifically.

CAR is distinct among other forms of action research due to its iterative nature, to continuously analyze and ultimately improve the steps of conducting and evaluating research. Therefore, we specifically focus on adopting and appropriately tailoring the CAR method within the general context of action research. Figure 4.1 provides an overview of the two step process involving:

- Definition of research objectives and scope followed by an iterative process.
- Diagnoses of the problem, planning the research activities, intervention, evaluation and reflection of the research results.

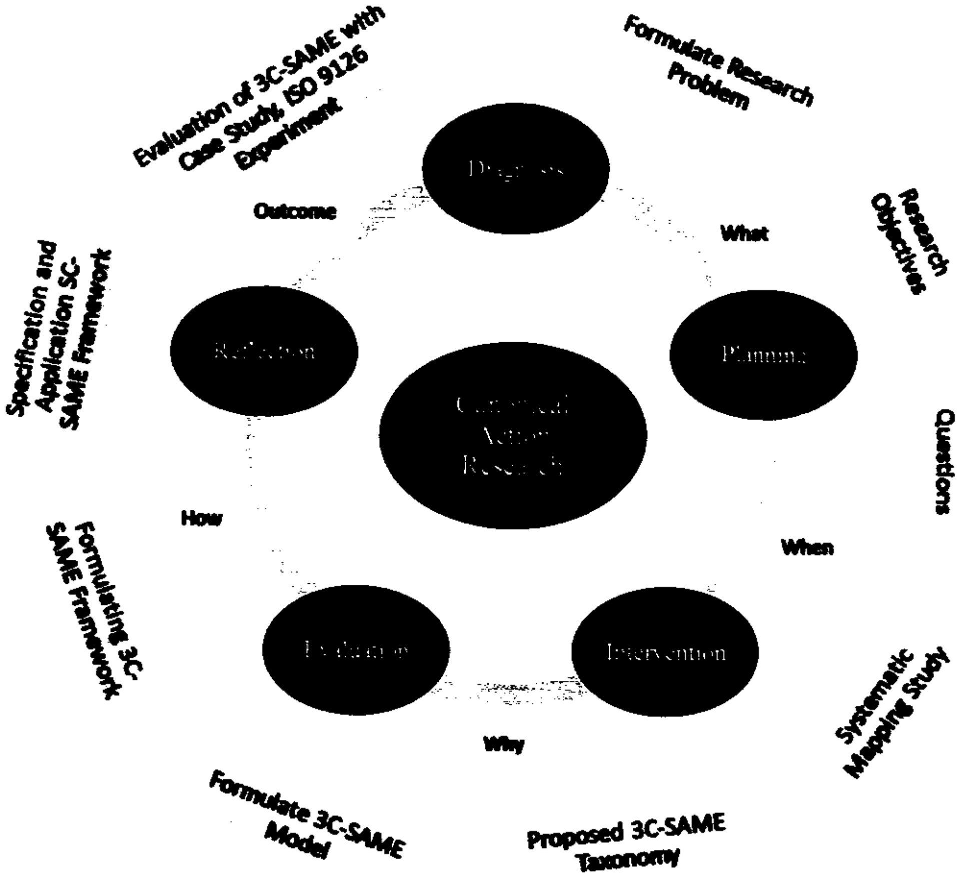

Formulate Research Problem
Research Objectives
Research Questions
Systematic Mapping Study
Proposed 3C-SAME Taxonomy
Formulate 3C-SAME Model
Formulating 3C-SAME Framework
Specification and Application 3C-SAME Framework
Evaluation of 3C-SAME with Case Study, ISO 9126 Experiment
Outcome
What
When
Why
How
Planning
Intervention

Figure 4.1: Overview of the Cyclic Process for Canonical Action Research

We have developed a detailed thesis of the problem statement and simultaneously move towards defining highest-level activities to solve the problem. This enables defining high-level objectives and scope of this thesis (step 1) in Figure 4.1, engaged in a cyclic process of problem diagnosing, so that planned research activities are always relevant to the problem and provide a rigorous evaluation of the solution (step 2) as illustrated in Figure 4.1. Step 2 is also referred to as the Cyclic Process Model (CPM). Once high-level objectives and scope of the study project are defined, we focus on executing the CPM for conducting the research along with evaluating the research results.

4.3.1 Diagnosing

Diagnosing refers to a systematic identification of the research problems that need to be addressed on a specific domain. We perform a literature review to identify the impacts, challenges and open issues in a structural way. The Systematic Mapping Review (SMR) is beneficial for evidence based impact of existing theories, framework and challenges of Agile GSD domain.

The problem-solution domain of our research includes challenges of global software engineering in a developing environment. However, we have specifically proposed a 3C-SAME framework. Therefore, the primary source of problem identification in our case is to review state-of the-research to identify the research problem that needs immediate solutions.

4.3.2 Planning

Planning involves steps and activities to conduct the research and to systematically develop the solution that aims to solve the identified research problem. In order to systematically quantify the research plan, we propose a framework formally known as ***3C-SAME***. The steps involved in the planning of ***3C-SAME*** is shown in Figure 4.2.

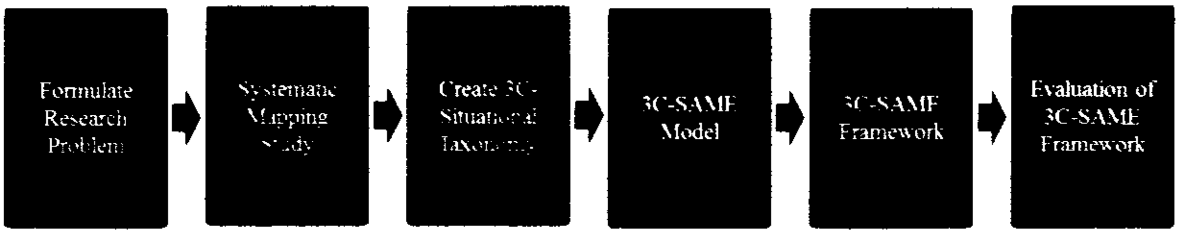

Formulate Research Problem
Systematic Mapping Study
Create 3C-Situational Taxonomy
3C-SAME Model
3C-SAME Framework
Evaluation of 3C-SAME Framework

Figure 4.2: Steps Involved in the ***3C-SAME*** Framework Planning

The proposed framework encapsulates a set of research activities to guide solution development. In addition, we determine the efforts required to evaluate the individual processes, activities and repositories of the ***3C-SAME*** framework. We presented and discussed the study plan that enables us to seek an early feedback from agile development and GSD community. The feedback was helpful to refine the solution and to derive plans for research validation.

4.3.3 Intervention

Intervention involves steps and activities to conduct the research results and to systematically document the solution that aims to solve the identified research problem. Based on the project plan, we conducted the research by considering individual processes and activities from *3C-SAME*.The first process includes challenges of GSE that allows us to systematically investigate and extract evolution of software industry. The second process includes agile practices that mitigate GSE challenges to evolve software development.

4.3.4 Evaluation of the Proposed Work

We have evaluated the proposed work through the following methods:

4.3.4.1 Evaluate the results, through ISO/IEC 9126-1 Quality Standard

We apply the ISO/IEC 9126-1 standard, which is internationally renowned for the evaluation of quality characteristics of a software product and the solution is utilized [108]. This evaluation aims to investigate the quality attribute of 3C-SAME framework and is based on an empirical investigation

4.3.4.2 Experimental Study and Case Study

The Proposed framework has been validated by implementing it in the real world project(s).

Based on the recommendation of the industry practitioners about the advancements and lacking of the proposed framework, desired changes were adjusted and incorporated within the framework.

4.3.5 Reflection

Reflection aims at an illustration of the impact of the research in terms of applicable results that provides research progression in a specific research community. After illustrating and evaluation of research contribution, we conclude with the reflection of research. In the context of our study, the appropriate reflection is based on empirical validation.

4.4 A Proposed (*3C*_Situational Agile Method Engineering for Global Software Development Environment): *3C-SAME* Framework

The framework represents the semantic structure of the system [109]. It is used to provide analysis of the real world concepts and is also applied for reusability, interoperability and customization [110]. According to Zachman's framework rules, no one can introduce a systematic methodology or working prototype without proposing a framework [66], which is providing a foundation for the system development. On the other hand, it is vital to mention Basic Generic Model would also prerequisite the element of a comprehensive framework [109]. Therefore, the proposed approach is composed of Basic Generic Model, and then this model shifts into Situation Specific Framework known as the *3C-SAME* Framework.

We propose a *3C-SAME* Model and *3C-SAME* framework, to overcome global software engineering challenges by applying agile philosophy. It has further enabled multiple stakeholders to work collaboratively in offshore such as USA and India, as well onshore such as Pakistan and India. In order to achieve this, we have identified a number of elements and situation factors from the literature as a corresponding and incorporated phase to the Model, as demonstrated in Figure 4.2 Model. This is particularly beneficial for establishing a conceptual base. The proposed Basic Generic Model is characterized of three major phases, i.e. pre-development process, development process and post-development process respectively. Every phase is further subdivided into tasks, artifacts and roles.

4.4.1 Conceptual Elements of *3C-SAME* Model

In the following, we present a logical arrangement of the core components of the *3C-SAME* Model, and it is formally known as conceptual elements of the Basic Generic Model. Each abstract element is elaborated beside its role in the framework and as shown in the Table

4.1 and in Figure 4.2. The theoretical roots of the key concepts and philosophical discussion of their meanings are omitted from this section, as the purpose of this study is to introduce the background and the key components of the framework. Furthermore, the processes describe what part of the model and activities were demonstrated in the model and how they were done. Generic view of the Model is illustrated in Figure 4.2 and brief summary of activities and processes shown in the Table 4.1.

Table 4.1: Summary of the Process, Activities and Repositories in the *3C-SAME* Model

Model processes	**Process Activities**	**Repositories**
Pre-development process	List of Backlog Items	(Knowledge Base Experience Factory such as Meta Model)
	Concept Definition	
	Prioritize Backlog Item	
	Finalized Backlog Item	
Development Process	Baseline Architecture Designing	Distributed white Board Concepts
	Distributed Sprint Development	
Post Development Process	Unit Testing	Tools such as Jira, TFS.
	Distributed Sprint Demo	
	Acceptance Testing	
	Handover	

(A) Processes in the Model: The processes characterize three different phases of the model Processes as shown in Table 4.1.

(I) Activities associated Processes: Each process comprises of a set of fundamental activities that depicted the distinct phases of agile software development and its solicitation in progression. Each of the pre-development, development and post development Processes are comprised of distinct activities as shown in Table 4.1.

(II) Role of Experience factory: Experience factory plays very important role along with the processes and activities as a source and coordinates the other elements of the model that contains instances from the requirement to handover of architecture.

(III) Processes and Activities in the *3C-SAME* Model: The model comprises of three main processes i.e. Pre-development, development and post-development. Each process further defines the set of distinct activities which aim to achieve the objectives of each process in an incremental and an iterative nature. In addition, a brief description of processes and activities allow us to highlight the research contribution, processes and activities as detailed in dedicated chapters of the thesis.

Process I- Pre Development Process:

This section describes the numerous pre development activities of our proposed *3C-SAME* **Model**.

- *Project Start Up*. The product owner is responsible for project start-up. He/she has the responsibility to authorize the charter for a particular project. He can use different tools (Incl. Jira, TFS, Scrum work and Wiki) to write requirements from user stories. This business process study provides business problem description, identification of solution substitutes while keeping cost constraints and associated benefits, thus providing the preferable alternate solution. Produced artefacts include a business case document that attracts sponsors for funding for project initiation.
- *User Stories.* Initial requirements (feature list) are termed as user stories which serve as input in the form of an index or story cards. The Initial user story card has been provided a base for the concept which is shifting into theme and this may be shifted into problem frame and outcome for standard backlog items.
- *Experience factory.* The repository includes *'experience factory'*. It consists of burn down charts that take user stories as input. Lean is one of the popular agile methodologies. Few concepts have characterized from lean for formulation of the framework. Lean practice; *waste elimination* has been applied to eliminating redundant requirements. Another lean practice associated with the repository is '*Plastic Bin*'. All the redundant requirements are shifting into plastic bins.
- *Backlog Planning.* In this phase, business realization is done in regard of its vision, review regarding objective criteria set for the project is conducted and produced deliverables are measured. Artifacts of this phase are project review document written by a technical writer, which provide approval to proceed to the next phase.
- *Product backlog.* It contains a prioritized feature list that contains a precise description of all desired functionality. Initially, estimates regarding required effort against each user

story are measured for implementing. It provides the GSD team in order to write clue against particular story's costs, which helps them to prioritize the user stories.

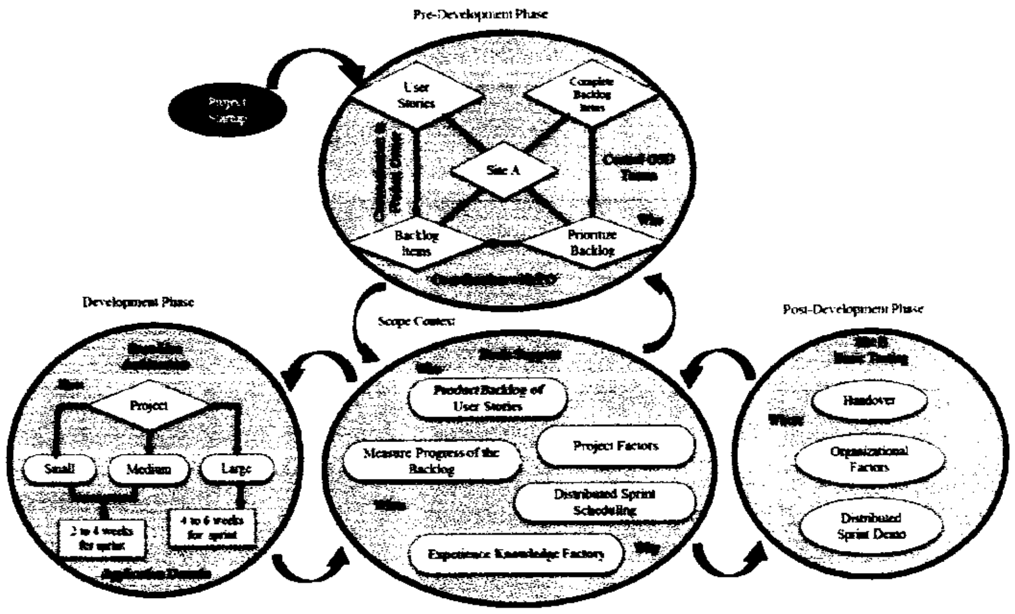


Figure 4.3: Proposed *3C-SAME* Model

- *Sprint Planning*. New and changed stories as the backlog grows are represented during a product backlog grooming meeting thus next sprint plans are also disclosed. Estimates also grow as the stories change, large user stories are also broken down into smaller stories. At the end, changes are communicated to each member to keep the product backlog updated.

Process II- Development Process:

In the development process following activities have been performed.

- *Architecture Construction*. This architecture is composed of system components and their hierarchy among the different components.
- *Distributed sprint planning*. This activity includes few selected tools such as a white board concept which is followed by sprint construction. Backlog item is used to determine the size and sprint suitable to the organization size like if the project size is small then it will take two weeks to complete sprint, but if the project is medium size, then it will extend for 4 weeks but for large projects it will exceed the 6 to 8 sprint time.

- *Sprint Construction.* In sprint development life cycle, this phase is sprint implementation.
- After sprint construction additional meetings are arranged if needed. Additional meetings cater to architectural and status meetings. The issues related to development are also discussed in additional meetings. In the development phase, collocated as well as distributed teams conduct the daily scrum meeting. After the completion sprint is demonstrated to onshore management that is followed by sprint retrospective. For next sprint it again goes to pre development phase in distributed scrum planning process. Sprint demo and sprint retrospective are achieved through communication platforms e.g., teleconferencing, office communication server, live chats and instant messaging, etc. Another lean role used in this phase is '*Last planner*'. Last planner is responsible for all the change management and communication of changes to the team directly or through communication platforms. '*Network analyzer*' is responsible for coordination within the team through the availability of a network.
- The team takes some spare time for reviewing current architecture during meeting sessions in order to plan and design forthcoming requirements. Daily stand-up meetings are organized in order to review past results and presents task updates and obstacles occurred. Sprint master is responsible for daily-up meetings which are usually held for 5-15 minutes. Afterwards sprint burn down chart is displayed and communicated publicly thus to remain on the track regarding progress. Meanwhile all the minutes, notification and important information are documented by the technical writer.
- Implementation: implementation is referred to as the most vital and sensitive phase. XP best practices have been applied like; pair programming, code ownership, continuous integration and code refactoring. As XP is link to automation therefore coding standards need to be followed in order to achieve good results using a sprint backlog thus improving the code structure. Pair programming provides a faster optimal solution by reducing defects rate, but is optional, depending on time and budget. Code refactoring is termed as internal structure changing without affecting the external behaviour of program structure, thus to reduce source code complexity and increase readability.

- Development Life Cycle: Sprint backlogs planning is responsible for extracting requirements for completing user stories. The team applies the scrum poker technique in order to calculate hours required to fulfil each task. After discussion on the initially selected user story team members pick their choice of card from the pool of product backlogs, thus representing their estimation regarding the work required for certain user story. Estimation units may vary, but there may be a number of hours or even days required for completing an associated task. Each team member defends their maximum or minimum estimations for updating the sprint backlog with updated information.
- Agile Social Network (ASNs) can be used to build an adoptable social network supporting the collaborations of GSE teams. ASNs are able to detect changes in the context and dynamically support different scenarios as per required. ASN's are formed on basis of collaborating network service application. For achieving a single network goal each ASN node collaborates with the other ASN node thus providing dynamic infrastructure i.e., adaptive towards changing context. ASNs provide the team round the clock productivity that could be supported by dynamically allocating collaboration between teams.

Process III- Post Development Process:

After the successful implementation of working prototypes, demonstrations to validate the prototype are carried out.

- *Sprint Testing.* A unit test has been designed before coding that will execute at this stage. Understanding feature requirements is necessary for developers in order to design associated test cases and the associated information regarding each system feature i.e., use test cases and detailed user story are provided to the development team so that no requirement is left unidentified and exceptional conditions are handled. As a result extensible codes and modularization is achieved. In sprint testing, unit testing is done in the source code and the smallest module is also tested. After the module passes unit testing it is considered successful and integrated before moving towards sprint review phase.

- *Sprint Review.* After sprint has been completed, the sprint review meeting is held in which team discloses the completed tasks. Team members are introduced with customers, stakeholders and the product owner in order to get the feedback and extract new requirements. New requirements are added to the product backlog along with incomplete requirements. After sprint review, sprint retrospective meetings are held in which team reviews their own work and prioritize the product backlog thus new improvements are made. The team also proposes concrete suggestion for the next sprint. After this phase, testing is done on the whole integrated and completed system. Specific system requirements are also evaluated. Relevant knowledge is required for writing and designing test cases, designing and writing test cases confirms the system from the stakeholder's point of view.
- *Hand Over.* This is the last phase in which, all artefacts like code, user guide, technical guide are handed over to user for release. Tutorial will be provided to user for better understanding and after deployment, system will be handed over to end user.

4.4.2 An Overview of the Proposed Situation Specific *3C-SAME* Framework

In the previous section, we presented the Basic Generic Model informally known as *3C-SAME* model to establish overriding GSE challenges by integrating Agile practices. We have characterized a Basic Generic Model, on and around the subject of agile methods in GSD environment for the development of computer based systems. As a result, we were able to describe that there is currently a dire need of the agile based framework in a situation specific perspective Agile GSD team. However, situational factors vary from situation to situation and are also characterized on the basis of situational factors such as *Application domain, Project organization and Organization development environment* which are often encountered by agile GSD practitioners and project managers. The focus of this section is on exploiting the need for effective, appropriate, and flexible *3C-SAME* Framework, in order to cater to the growth in the complex situations for distributed development.

Moreover, such type of situational methodologies and their associating elements are employed to reduce time and cost, and to improve quality and provide an ease of use facility to Practitioners. Furthermore, we establish a new paradigm for agile GSE organizations in order to improve the successful development strategy which is helpful in contextual situations to generate situational methods.

As the time progresses, the growth of the software industry has increased tremendously. This multifaceted growth can be attributed to the fact that the software industry has faced challenges such as diverse project nature [111]. Consequently, the need for well-defined guidelines is very crucial to fulfill the demands of current computer-based systems. Furthermore, in these circumstances it is crucial for practitioners to find a comprehensive development method that provides efficient and effective results for agile teams in a distributed development environment.

Now the question arises about the tailoring of an agile method which provides opportunities for the distributed teams to overcome the different situations during software development. In order to provide customization facility, Situational Method Engineering (SME) would establish a single universally applicable methodology. This discipline (SME) is applicable to formulate the construction and adjustment of methodologies to specific situations [113].

It is used for developing software methodologies that are tailored to fit the specific circumstances of the GSD project situation. Hence, SME aims at providing method construction techniques by reusing existing methods, practices, techniques and approaches as shown in the Figure 4.3. The newly formulated methods are more flexible and easier to adapt according to the situations [21, 55,115].

Under the guidelines of SME "the field to make task-specific methods, called situational methods, from components of the existing methods, called methods fragments", various terms were used as building blocks for various approaches such as fragments, chunks and patterns [113, 114].

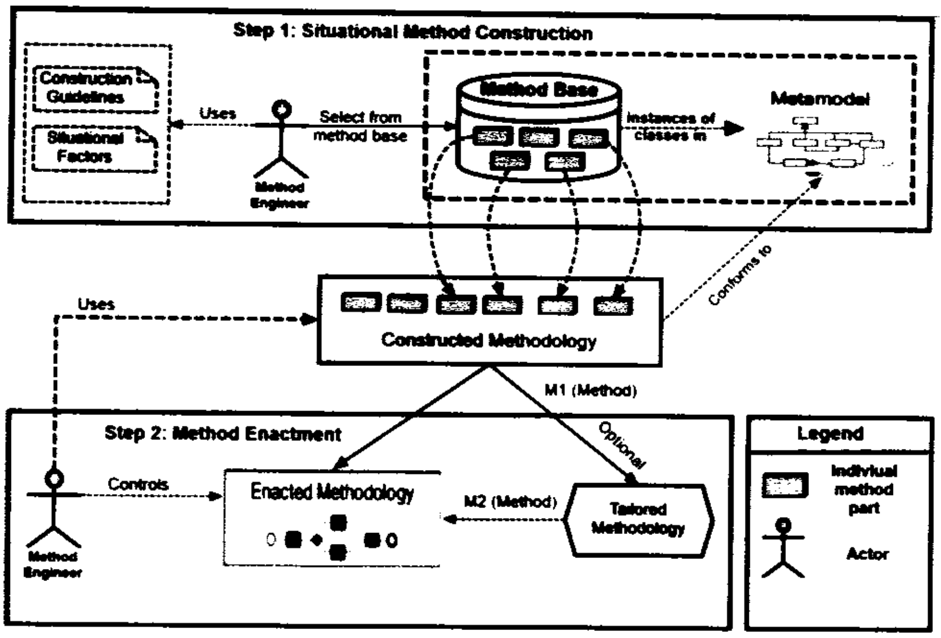


Figure 4.4: Situational Methods Engineering Work Fflow [68]

4.4.3 A Proposed *3C-SAME* Framework and its Capabilities

The *3C-SAME* Situation Specific Framework presented in this section, represents both Agile practices with situational method, and a process for GSD. The *3C-SAME* Situation Specific Framework for the entire methodology as shown in Figure 4.4, provides an overview of the proposed framework. Furthermore, it illustrates four layers, namely the characterization of Situations, constructed situational methodology, assembly of the method Fragments and Agile Team coordination method execution respectively in order to support situational method engineering activities including representation, retrieval, enactment and composition. These individual elements of the *3C-SAME* Situation Specific Framework are subsequently detailed in the following subsections.

Phase 1: Situation Method Formulation

The first phase of the *3C-SAME* Situation Specific Framework is characterized of Situations. At this point the situational elements of the work are identified, in order to imply the formulation of the method to be employed. There are many ways to describe a Global Agile Method Experience factory, which is used for information and repository of integrated experience to meet the needs of Agile methods, established on the specific situation. One recommendation has been to include Meta Model for specifications and identification of concepts for specific situations.

Phase 2: Constructed Situational Methodology

The second phase of the ***3C-SAME*** Situation Specific Framework is constructed situational methodology. The appropriate method fragments are selected from the Method Repository and it is also assembled into method components, and then stored in Agile method experience factory for future reference when required. These fragments are then organized and sequenced according to the organization of the framework. These transactions are referred to collectively as the constructed situational methodology, which is an executable method that can be enacted. Contextual Analysis support has been included in the selection of possible contextual factor against the situation.

Phase 3: Assembly of Method Fragments

The third phase of the ***3C-SAME*** Situation Specific Framework is an Assembly of Method. It is also helpful to reduce the method engineer's manual burden with the selection of appropriate method fragments. Once the appropriate method fragments have been selected and assembled according to contextual factors, the method may be enacted and structured according to the situation.

Phase 4: Agile Team Coordination Method Execution

The Fourth and final phase of the ***3C-SAME*** Situation Specific Framework is Agile Team Coordination Method Execution. Once the fragment has been selected for the appropriate method, which is also constructed in accordance with the help of contextual factors, the result is the method management being stored in the method composer. The outcome of this method composer is the Situational Agile Method.

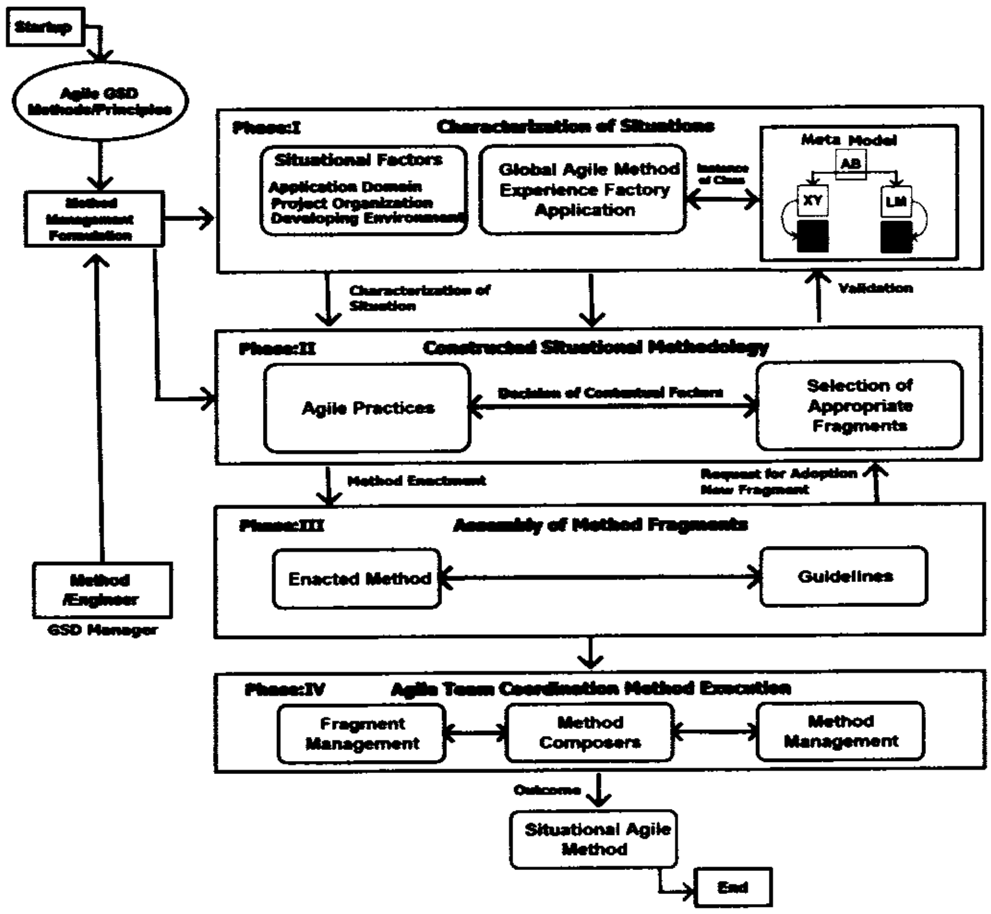

Startup
Agile GSD Methods/Principles
Method Management Formulation
Method /Engineer
GSD Manager
Phase:I
Characterization of Situations
Situational Factors
Application Domain
Project Organization
Developing Environment
Global Agile Method Experience Factory Application
Instance of Class
Meta Model
AB
XY
LM
Characterization of Situation
Validation
Phase:II
Constructed Situational Methodology
Agile Practices
Decision of Contextual Factors
Selection of Appropriate Fragments
Method Enactment
Request for Adoption New Fragment
Phase:III
Assembly of Method Fragments
Enacted Method
Guidelines
Phase:IV
Agile Team Coordination Method Execution
Fragment Management
Method Composers
Method Management
Outcome
Situational Agile Method
End

Figure 4.5: A Proposed *3C-SAME* Situation Specific Framework

4.5 Summary of the Chapter

The lightweight *3C-SAME* Framework for Agile GSD presented in this chapter, provides a number of potential benefits over existing ones in order to respond to the research questions *RQ 2: What types of Agile solutions have been suitable for GSD and their impact on GSD Challenges?*

RQ 3: What is the state of key components of the situational frameworks which have been provided to situational methods as solutions?

It is a flexible method in that it provides guidelines for each phase of the framework, and the capability to customize and tailor the Agile situational methods based on distributed environment. The Proposed *3C-SAME* Framework has the usability feature, suited to the novice software engineer, as our situational method does not require significant expertise and is therefore particularly suited to novice software engineers. In addition, the proposed *3C-SAME* Framework is particularly useful in projects lacking a defined software development method. Furthermore, it is used independently upon the utilization of any other systems development process.

As part of the framework, we have presented a number of novel concepts, including that of a 'method composer', 'fragment management', and 'method management' methods that provide the team coordination between software engineers and GSD managers for the specific project at hand. Furthermore, this framework provides a foundation and prototype-based tool support that enables agile distributed development.

In the next chapter we will present the *3C-SAME* framework into work, two industrial case studies, which embody and enhance the *3C-SAME* Framework. The empirical evaluations of the *3C-SAME* framework along with its specification and application of the *3C-SAME* Framework (Chapter 5) combination by way of a case study, will then be presented in chapter 6.

Chapter 5: Specification and Application of the *3C-SAME* Framework

5.1 Chapter Overview

The *3C-SAME* framework has been presented in the previous chapter. In addition, the proposed framework also provides a flexible, lightweight, iterative approach, in agile software development practices for GSD environment, based on situational method engineering principles.

In this chapter, we have characterized the specification of the *3C-SAME* framework, as a prototype application for the validation of situational methods. The point is to draw the fact that prototype support for framework is an important and effective way of proving framework (conceptual work) into practice, which is more practical and easier to use for software industry practitioners.

The overriding objective of the current chapter is to provide the designing and application details of the *3C-SAME* framework. In addition, the current chapter also provides the Meta model description, information related to architecture, prototype specification and screen shots of the prototype. Furthermore, our *3C-SAME* framework may also provide the facility to Agile GSE practitioners to improve the process of appropriate task selection and customization according to the situation to support team coordination. The evaluation of the *3C-SAME* framework with the help of real world case studies, ISO 9126-1 Model and also from experimentation from software industry participants is provided in the subsequent Chapter 6.

5.2 Prototype Support for the *3C-SAME* Framework

The application of *3C-SAME* framework is therefore to prove that it is possible to shift the proposed *3C-SAME* framework into working prototype in order to facilitate the Agile GSD managers and practitioners. In addition, the prototype is also utilizing ISO/IEC 24744 meta-model for the formal and standard definition of method fragments. Furthermore, it supports assembly-based method engineering guidelines [68]. The working prototype also provides interfaces to define situational factors at fragment level. Based on these situational factors, practitioners can easily retrieve fragments during the composition of a new agile method. It utilizes a multi fragment selection for the fragment retrieval process [114]. GSD managers and practitioners can also define new situations and new fragments as required for method management.

The design and construction of *3C-SAME* prototype has been impacted by numerous constraints. The most important constraint which has the highest impact, is the time limitation imposed by the schedule of study. The first version of the working prototype has been developed with fundamental features, to demonstrate to and get feedback from practitioners of the software industry. They have evaluated the working prototype and suggested a few valuable changes. Based on their feedback, working prototype has been further reviewed with more focus on required functionality and towards its situation specific nature. The detail description of the final version of working prototype which is formally known as *3C-SAME* prototype is described in the following sections.

5.3 *3C-SAME* Prototype Specifications

The *3C-SAME* prototype is established by a series of specifications that should provide significant functions about Agile GSD teams in a situational context. At an abstract level prototype specification is as under:

- This information involves improvement in the existing situation specific Agile GSD practices in terms of tailoring and adoption;
- Mitigate the renowned challenges such as customization according to the situation in order to overcome team coordination faced by Agile GSD managers and practitioners during development;
- Provide a situational framework based approach for the appropriate selection of Agile fragments from Agile methods;

The fundamental detail level prototype specifications are also given below:

- To create and modify different kinds of agile fragments which conform to ISO/IEC 24744 meta model;
- To define the relationships between fragments;
- To define the context for fragments at fragment level;
- To define the software development project situations in terms of organizational characteristics, project characteristics and application characteristics;

- To retrieve the suitable fragments out of method base for project situation;
- To define the coherent methodology based on selected fragments;
- To create and modify contextual factors such as Project Size;

5.4 Application of Meta-Model for *3C-SAME* Framework

Meta-models are used as specifications and identification of concepts and their relationships between the different objects [115]. Metamodels are used in order to apply geeneral purpose development approaches such as object-orientation, component-based development and agent-orientation [115]. In both software engineering and model-driven development domains a meta-model has two type of syntexes [64, 65] which is given below.

Abstract Syntax: Identifies the concepts and their possible relationships between those elements specified by certain language which is shown in Figure 5.1.
Concrete Syntax: Defines notations that contributes to the representation and development of models in certain language.

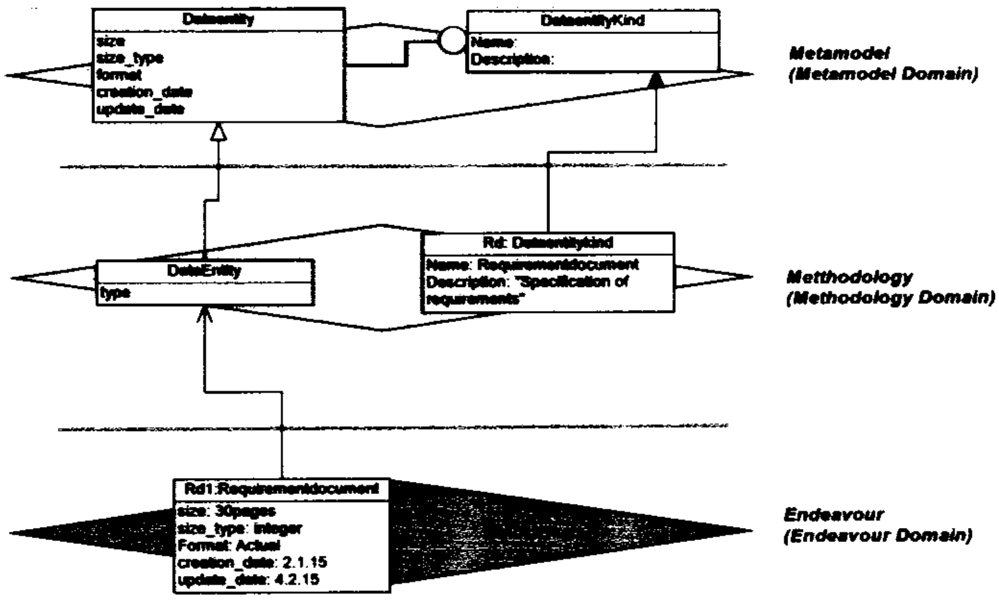


Figure 5.1: Abstract View of Meta Model

We have adopted a metamodel for specification and composition of *3C-SAME* framework in order to define the concepts and their relationship.

5.4.1 Meta Model ISO/IEC 24744

The application of the framework is based on concepts and their associations. In our context, we have preferred ISO/IEC-24744-SEMDM Meta model [68] shown in Figure 5.2, in order to formulate a framework for the tailoring and customization of Agile GSD methodologies. Despite the differences in finer details, all agile methodologies follow a paradigm [117], which is more often represented by a Meta model. Therefore, a Meta model is a used for identification of concepts and their associations [116]. According to the best of our knowledge and literature, the most recent work in order to formulate the specification has been used in ISO/IEC 24744 Meta model and resolves their all known problems such as software specification along with their associative knowledge repositories.

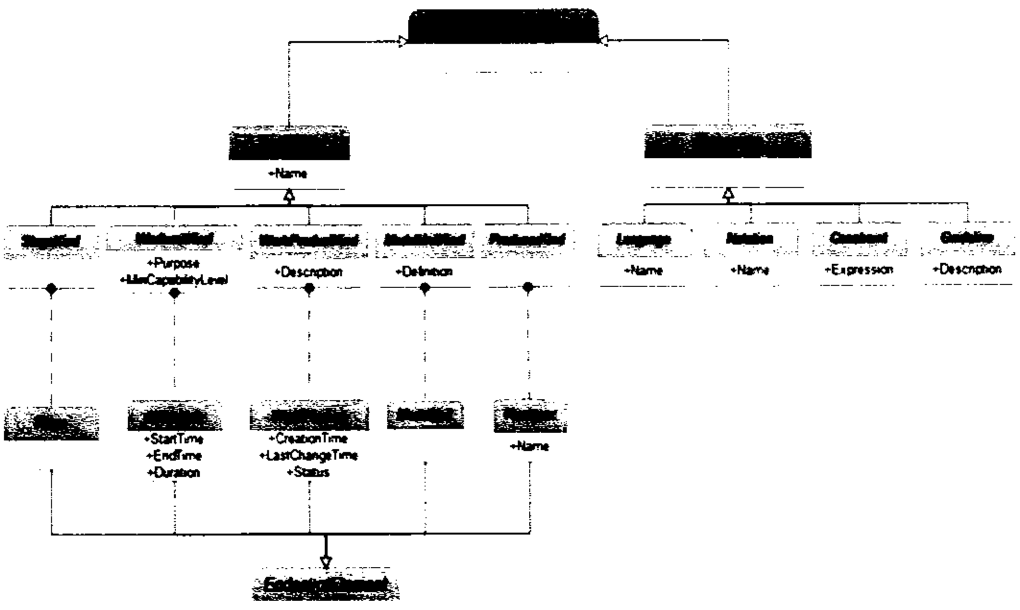


Figure 5.2: ISO/IEC-24744-SEMDM Meta model

In addition, it also guides different systems and processes to generate the work product. Furthermore, it also assures all aspects of methodologies i.e. processes to be followed, creation of work products and the people who follow the process to generate the work products at depth which other Meta models don't address.

The core classes of the Meta model describe and represent agile methodology fragments in a standard way so that composition of different fragments derived from different agile methodologies could become easier. The detailed explanation of ISO/IEC 24744 [65] meta-model core classes is presented in Table 5.1.

Table 5.1: Meta-model core classes

Name	Super Class	Description
Methodology Element	Element	An abstract class representing elements created by the method engineer in the construction of new methodology or tailoring the existing one. It is a high level class which is further specialized into the Template and Resource.
Endeavour Element	Element	An abstract class representing any element created by developer while using any particular methodology defined by method engineer. It is a high level class which is further specialized into Stage, WorkUnit, WorkProduct, ModelUnit, and Producer.
Template	MethodologyElement	An abstract class representing methodology elements which will be instantiated by developers while performing in any particular endeavor. It is a high level class which is further specialized into StageKind, WorkUnitKind, WorkProductKind, ModelUnitKind, and ProducerKind.
Stage Kind	Template	An abstract class representing different kinds of stages in a particular methodology. It is part of a process aspect of the methodology. For example, Exploration, Iteration, Productionizing, Maintenance, and Death stage kinds in Extreme Programming.
Stage	EndeavourElement	An abstract class to represent managed time frame within an endeavour. For example, during a particular software development project using Extreme Programming agile methodology product is developed in multiple iterations (Iteration 1, Iteration 2 etc.). These iterations are instances of Iteration class defined in methodology domain.
Work Unit Kind	Template	An abstract class representing different kinds of work units in software development methodologies which has the specific purpose of the endeavor domain. It is further specialized into ProcessKind, TaskKind and TechniqueKind. Write-Code (Task-Kind), Pair Programming (TechniqueKind) in Extreme Programming.
Work Unit	EndeavourElement	An abstract class to represent jobs performed during a software development project. It is further specialized in Process, Task and Technique. For example, in a particular project Programmer A and Programmer B is writing code for a particular feature of software product being developed

		together utilizing pair programming technique defined in the Extreme Programming agile methodology.
Work Product Kind	Template	An abstract class represent different kind of work products in software development methodologies which have characterized based on nature of its content and purpose behind its usage in the endeavour domain. It is further specialized into DocumentKind, ModelKind, SoftwareltemKind, HardwareItemKind and CompositeWorkProductKind. For example, Iteration Plan in Extreme Programming.
Work Product	EndeavourElement	An abstract class representing an artifact of interest in the endeavour domain. It is further specialized into Document, Model, SoftwareItem, HardwareItem and CompositeWorkProduct. For example, during a particular software development project Iteration Plan has been built for Iteration1.
Model Unit Kind	Template	A concrete class to represent a different kind of model units used in software development methodologies categorized by nature of the content it represents. For Example Class, Association model unit kinds in a Class Diagram.
Model Unit	EndeavourElement	A concrete class to represent atomic units of a model. For example, a class diagram representing structural aspects of a particular software product contains a number of classes, associations between them and attributes defined in classes are all model units.
Producer Kind	Template	An abstract class representing different kind of producers working in a software development methodology. Producers kinds are defined based on area of expertise. It is further specialized into RoleKind, TeamKind and ToolKind. For Example, Programmer, Customer, Tester, Tracker, Coach, Consultant, Big Boss in Extreme Programming.
Producer	EndeavorElement	An abstract class representing an agent who executes the work units in an endeavor to produce work products. It is further specialized into d into Role, Team, Person and Tool. For Example, during a particular software development project Programmer A is an instance of Programmer class defined in methodology domain.
Resource	MethodologyElement	An abstract class representing methodology element which is used directly at endeavour domain. There is no instantiation process required to use it at endeavour level. Any guideline or reference used in methodology could be

		represented by this class. It is further specialized into Language, Notation, Constraint, and Guideline.
Guideline	Resource	A class to represent guidelines which guides the developer about how a particular fragment could be used in endeavor domain.
Language	Resource	A class to represent the structure of different model unit kinds that focus on particular modeling perspective of the subject at hand. For Example, A Class Modeling Language to model structural aspect of a software product.
Constraint	Resource	An abstract class to represent condition that holds at a certain point in time. It is further specialized into PreCondition and PostCondition. For example, Iteration execution should start when an Iteration Plan is in completed state.
Notation	Resource	A class to represent graphical syntax that can be used to depict models created using a language. For Example, Class Diagram.

5.4.2 Agile Fragment

In literature, a small portion of the methodology is termed differently, but the two most common terms are Method Fragment [118,119, 120] and Method Component [121]. We have adopted the term Fragment for our proposed framework in accordance with SEMDM and more specifically Agile Fragment. The main sources of agile fragments are existing agile methodologies like Scrum and XP. These fragments are extracted from their primary sources as per standard format in order to make them compatible with other fragments while constructing new agile methodology. For ensuring this compatibility we may also conform to classes defined in a standard metamodel. Metamodel also becomes the second source for creation of fragments by instantiating a standard metamodel element. As we have mentioned in previous section, we have used ISO/IEC 24744 metamodel to formalize agile fragments. There are three main kinds of agile fragments which play a vital role for the tailoring of any Agile methodology: work unit kind, work product kind and producer kind.

The typical role kinds participating in XP and Scrum along with their responsibilities have been briefly presented in Table 5.2.

Producer Kind fragments produce Work Product Kind fragments by performing different kind of Work Unit Kind fragments. For example, in Extreme Programming two programmers (Both instances of Producer Kind) produce code (Work Product Kind) by writing code (Work Unit Kind) using the pair programming technique (Work Unit Kind).

Table 5.2: Agile Role Kind Fragments

Name	Source	To Class	Responsibilities
Customer	XP	Agile	User representative who write user stories, make decisions, and write functional tests.
Programmer	XP	Agile	An individual who is responsible to design application, write unit tests, write code, and refactor code.
Tester	XP	Agile	Tracker is an individual who keeps an eye on progress of project and notify the team and other stakeholders about the project progress. Who also keeps log of functional test scores and defects reported.
Coach	XP	Agile	An individual who is responsible for the whole process, notice people deviation from team's process and guides the team about process.
Consultant	XP	Agile	An individual who helps the team when they need consultancy in deep technical knowledge.
Big Boss	XP	Agile	An individual who helps the team to resolve their issues like they need a human resource, financial resource.
Product Owner	Scrum	Agile	Product owner is a user representative who is responsible for maintaining the requirements and prioritizing the requirements. He also responsible to keep the track of values to the business based on these requirements. Project success or failure totally depends on this producer.
Scrum Master	Scrum	Agile	Scrum Master is a facilitator who is responsible of coordination between team, to help them in the impediments being faced during the development.

5.4.3 Agile Methodbase

The repository of method parts (method fragments, chunks or components) is called 'method base' [20,118]. The construction of agile methodologies is possible if we have a comprehensive repository of agile method components known as experience factory. The ultimate goal of the framework is to construct agile methodologies based on different existing fragments of agile methodologies. Therefore, the definition of each fragment also conforms to standard development as we have defined in the previous section in the form of metamodel. In the extraction of the agile fragments, we have used two reputable agile methods Scrum and XP [118]. In this study, we have demonstrated our approach using these two methods. In the future, we can add more fragments to the repository so that a vast variety of agile methodologies could be constructed based on different situations. Different kind of agile fragments document kind is presented in Table 5.3.

Table 5.3: Fragment Document-Kind

Name	Source	To Class	Description
Release	XP, Scrum	Agile	Software part delivered to customer at the end of iteration.
Final Product	XP, Scrum	Agile	Final product delivered to customer after completing much iteration.
Story Cards	XP	Agile	It specifies which user stories are being developed in which release of the system along with release dates.
Task Cards	XP	Agile	Task card related to a specific user story.
Product Backlog	Scrum	Agile	Master list of functionality required in the product to be developed.
Sprint Backlog	Scrum	Agile	List of tasks which the scrum team committed for a particular sprint.
User Manuals	XP, Scrum	Agile	A guide for the user to explain different parts of the software product delivered to the customer.
Release Burn down Chart	Scrum	Agile	Team progress chart against release plan.

Sprint Burn down Chart	Scrum	Agile	Graphical representation of the remaining work over the sprint duration.
Task Board	Scrum	Agile	A board which shows all the tasks being implemented by the team in a particular sprint.
Unit Test	XP	Agile	Written at class level and only for public interfaces. For each class in the system, a unit test exists.
Production Code	XP, Scrum	Agile	An executable specification of the product being developed.
Iteration Plan	XP	Agile	A list of user stories and its associated tasks to be worked on in the iteration.
Functional Test	XP	Agile	Written by customer to specify the scenarios to test a particular user story.

5.5 *3C-SAME* Framework Architecture

The ***3C-SAME*** Framework architecture is composed of three layers: *presentation, business logic and data access layer, respectively*, which are shown in Figure 5.2. The presentation layer is responsible for getting inputs from method engineers (i.e. Agile fragment definition, project situational characteristics) and transforms it into data structures which can be parsed by business logic layer.

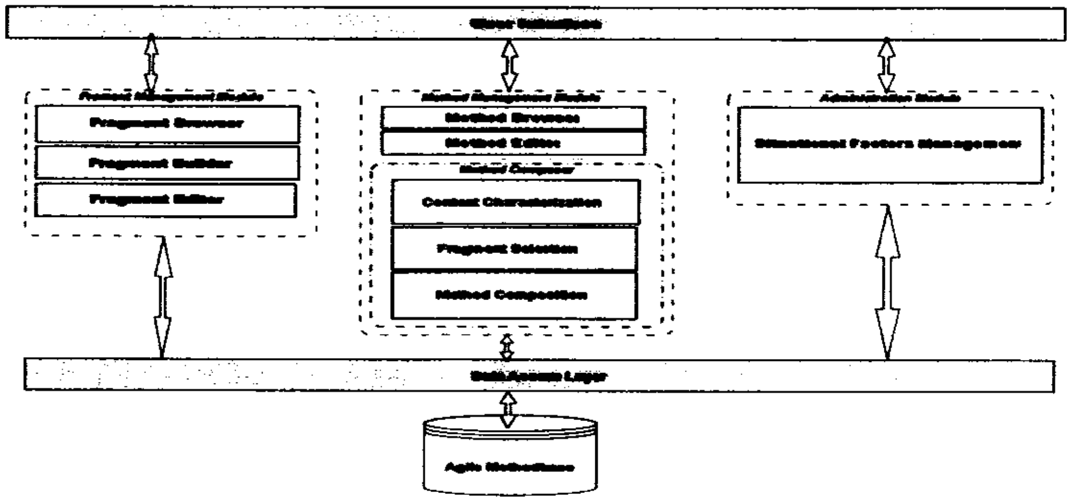


Figure 5.3: ***3C-SAME*** Framework Architecture

The data access layer also provides the facility to access the Agile Methodbase and provide functions for querying, modifying, creating, and deleting method fragments.

Business logic layer is divided into three modules such as Fragment Management Module, Method Management Module and Administration Module respectively.

Fragment Management Module is responsible to create, modify and search the fragments in conformance with ISO/IEC 24744 metamodel. This layer is also responsible for defining relationships between fragments in order to build situation. The Method Management module is responsible for the composition of situational agile methods based on situational factors provided by GSD managers and practitioners. It supports the method fragments which is suitable for a particular project situation. Administration module is responsible for performing some administration tasks. Currently it supports only one sub-module called Situational Factor Management which defines, modifies and browses situational factors.

5.6 Components of *3C-SAME* Prototype

The *3C-SAME* prototype is constituted by a following series of components that is helpful in order to prove framework (theory) into practice.

5.6.1 User Interfaces for *3C-SAME* Prototype

The *3C-SAME* prototype User Interface (UI) enables the method engineer to navigate through different features of the system. The home interface is divided into three components

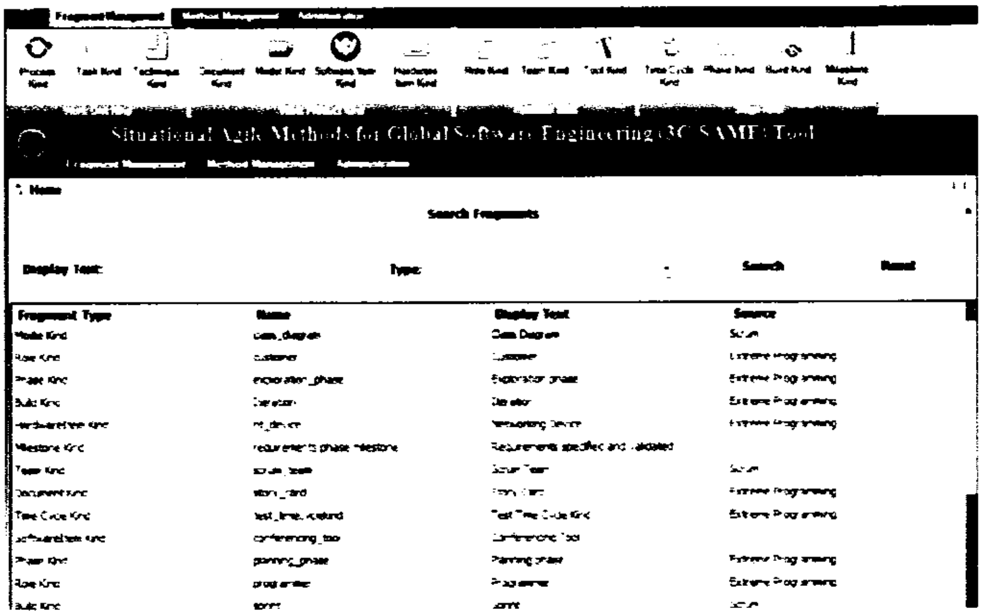
(i) Fragment Management

(ii) Method Management

(iii) Administration.

Fragment management menu provides access to all user interfaces in order to create different kinds of fragments as shown in Figure 5.3. This menu is further divided into process kind, task kind, and technique kind respectively in sub menus.

In the preceding sections these three items are described in detail. The remaining area of the prototype has been used to show different user interfaces in a tabular format. It lists down all the existing agile fragments into the fragment repository. The user can filter out fragments by name and by type on this screen. User can open any fragment in edit format by double clicking on the interested fragment row in the grid. All the updates to fragments are built using the Home interface.

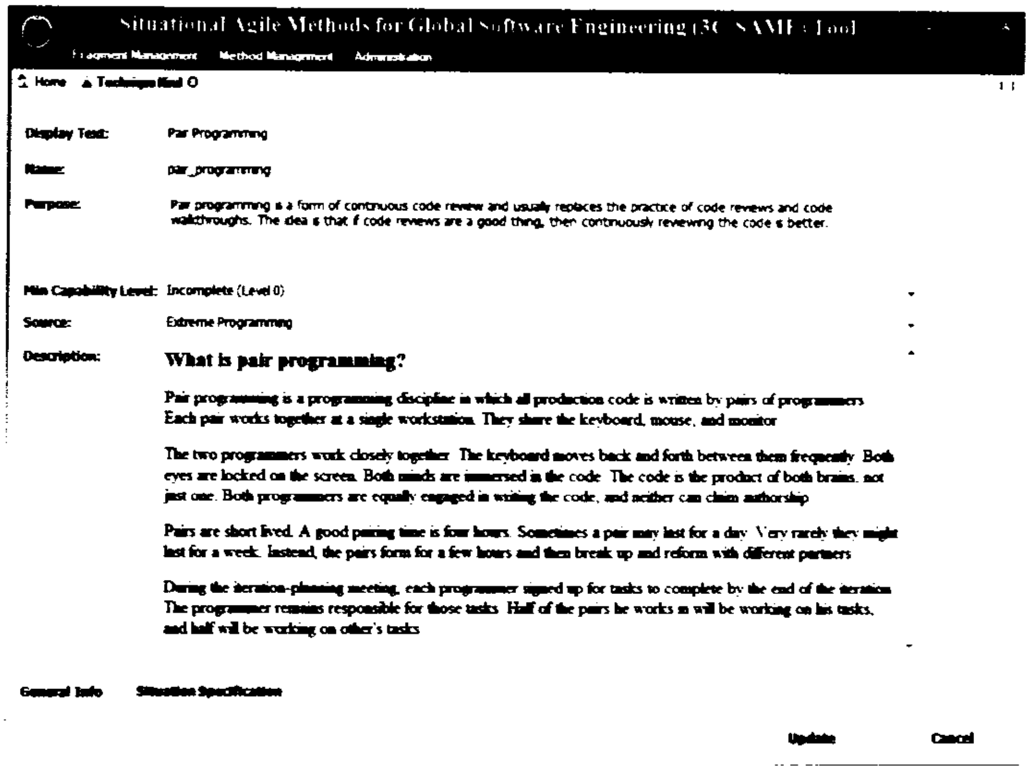


Fragment Type	Name	Display Text	Source
Model Kind	class_diagram	Class Diagram	Scrum
Role Kind	customer	Customer	Extreme Programming
Phase Kind	exploration_phase	Exploration phase	Extreme Programming
Build Kind	iteration	Iteration	Extreme Programming
HardwareItem Kind	nt_device	Networking Device	Extreme Programming
Milestone Kind	requirements phase milestone	Requirements specified and validated	
Team Kind	scrum_team	Scrum Team	Scrum
Document Kind	story_card	Story Card	Extreme Programming
Time Cycle Kind	test_time_cycle	Test Time Cycle Kind	Extreme Programming
SoftwareItem Kind	conferencing_tool	Conferencing Tool	
Phase Kind	planning_phase	Planning phase	Extreme Programming
Role Kind	programmer	Programmer	Extreme Programming
Build Kind	sprint	sprint	Scrum

Figure 5.4: *3C-SAME* Prototype 'Home' Interface

Work Unit Kind: This is to create Process Kind, Task Kind, Technique Kind and Task-Technique Mapping Kind fragments. Figure 5.5 has shown the interface to define 'Technique Kind' tab and 'Situation Specification' tab respectively.

- **Work Product Kind:** To create Document Kind, Model Kind, Software Item Kind, Hardware Item Kind and Composite Work Product Kind fragments.
- **Producer Kind:** To create Role Kind, Team Kind and Prototype Kind fragments.
- **Stage Kind:** To create Time Cycle Kind, Phase Kind, Build Kind and Milestone Kind fragments.
- **Support:** To create Corporation, Source and Reference items to support all kind of fragments.



Display Text: Par Programming

Name: pair_programming

Purpose: Pair programming is a form of continuous code review and usually replaces the practice of code reviews and code walkthroughs. The idea is that if code reviews are a good thing, then continuously reviewing the code is better.

Min Capability Level: Incomplete (Level 0)

Source: Extreme Programming

Description: **What is pair programming?**

Pair programming is a programming discipline in which all production code is written by pairs of programmers Each pair works together at a single workstation. They share the keyboard, mouse, and monitor

The two programmers work closely together The keyboard moves back and forth between them frequently Both eyes are locked on the screen. Both minds are immersed in the code The code is the product of both brains, not just one. Both programmers are equally engaged in writing the code, and neither can claim authorship

Pairs are short lived. A good pairing time is four hours. Sometimes a pair may last for a day Very rarely they might last for a week. Instead, the pairs form for a few hours and then break up and reform with different partners

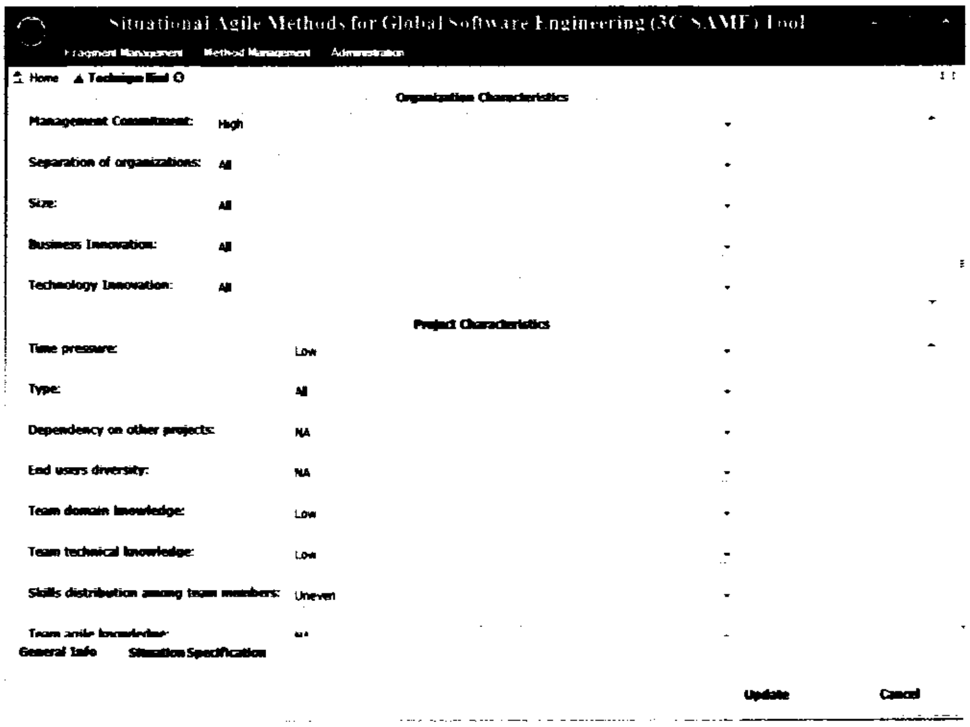
During the iteration-planning meeting, each programmer signed up for tasks to complete by the end of the iteration The programmer remains responsible for those tasks Half of the pairs he works in will be working on his tasks, and half will be working on other's tasks

Figure 5.5: 'General Info' Tab on Technique Kind Fragment Interface

5.6.2 Method Management Menu

Method management menu provides access to all user interfaces which are required to construct situational methods and already created methods. This menu is further divided into two sub menus:

- **Method Composer:** This menu item gets situational factors input from the method engineer to define the requirements for the method to be built. Figure 5.6 has shown interface of Method Composer to capture the method requirements of the method engineer.
- **Methods:** This menu item provides access to the screen which list down the all situational methods created using the prototype.



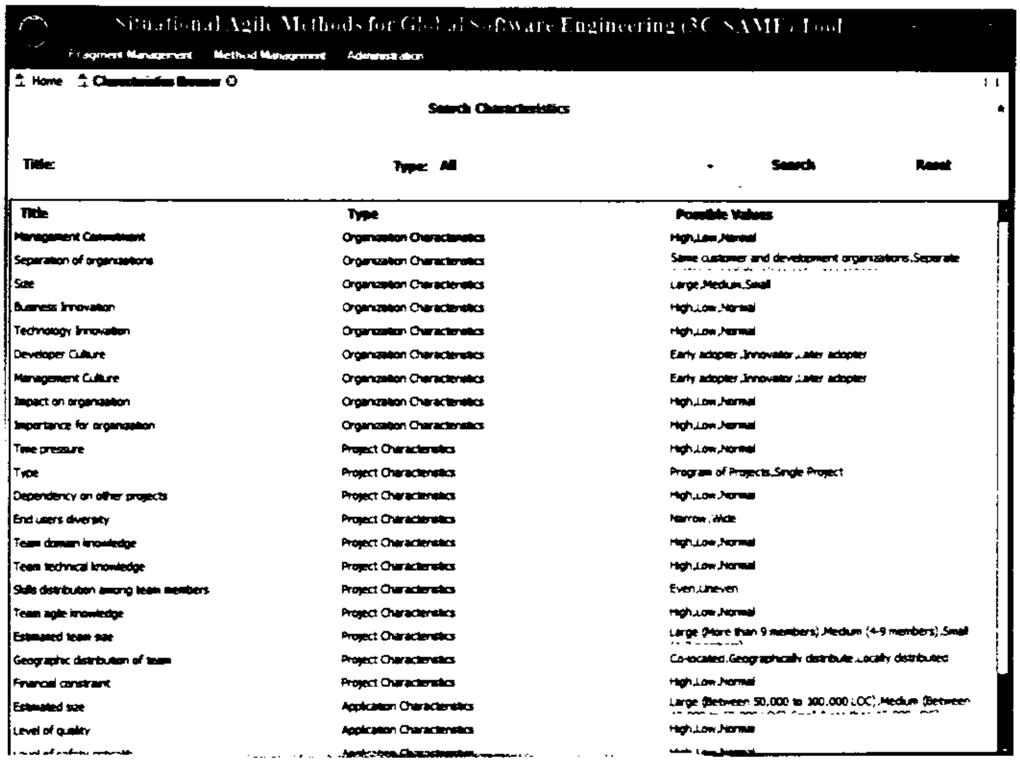
Management Commitment:	High
Separation of organizations:	All
Size:	All
Business Innovation:	All
Technology Innovation:	All
Time pressure:	Low
Type:	All
Dependency on other projects:	NA
End users diversity:	NA
Team domain knowledge:	Low
Team technical knowledge:	Low
Skills distribution among team members:	Uneven
Team agile knowledge:	[illegible]

Figure 5.6: 'Organization and Project Characteristics' Interface of Method Composer

5.6.3 Administration Menu

Administration menu provides access to all user interfaces which are required to define different lookups used in the system. Currently this menu contains only two menu items which are as follows:

- **Characteristic Browser:** This menu item gives access to the user interface which lists down all the situational factors defined in the system. Method Engineer can add and update possible values defined for each factor. This menu item helps to define new situational factor to be used in method composer and at fragment situation specification.


Situational Agile Methods for Global Software Engineering (3C-SAME) Tool

Fragment Management Method Management Administration

Home Characteristics Browser

Search Characteristics

Title: Type: All Search Reset

Title	Type	Possible Values
Management Commitment	Organization Characteristics	High,Low,Normal
Separation of organizations	Organization Characteristics	Same customer and development organizations,Separate
Size	Organization Characteristics	Large,Medium,Small
Business Innovation	Organization Characteristics	High,Low,Normal
Technology Innovation	Organization Characteristics	High,Low,Normal
Developer Culture	Organization Characteristics	Early adopter,Innovator,Later adopter
Management Culture	Organization Characteristics	Early adopter,Innovator,Later adopter
Impact on organization	Organization Characteristics	High,Low,Normal
Importance for organization	Organization Characteristics	High,Low,Normal
Time pressure	Project Characteristics	High,Low,Normal
Type	Project Characteristics	Program of Projects,Single Project
Dependency on other projects	Project Characteristics	High,Low,Normal
End users diversity	Project Characteristics	Narrow, Wide
Team domain knowledge	Project Characteristics	High,Low,Normal
Team technical knowledge	Project Characteristics	High,Low,Normal
Skills distribution among team members	Project Characteristics	Even,Uneven
Team agile knowledge	Project Characteristics	High,Low,Normal
Estimated team size	Project Characteristics	Large (More than 9 members),Medium (4-9 members),Small
Geographic distribution of team	Project Characteristics	Co-located,Geographically distribute,Locally distributed
Financial constraint	Project Characteristics	High,Low,Normal
Estimated size	Application Characteristics	Large (Between 50,000 to 100,000 LOC),Medium (Between
Level of quality	Application Characteristics	High,Low,Normal
Level of safety [illegible]	Application Characteristics	High,Low,Normal

Figure 5.7: 'Characteristic Browser' Interface

5.7 Analysis with Existing Related Studies

The following section represents the comparison with the *3C-SAME* prototype based approach. Brief description of the underlying related study has been presented in given below. To construct and manage situation specific methodologies, Computer Aided Method Engineering (CAME) prototype also provides support to the method engineer [123]. These kinds of prototypes are very useful in automating the process of method engineering. Practitioners can adopt various software development strategies and reuse existing methodologies [123]. Furthermore, keeping in view the situation of software development, CAME tool also plays a vital role to provide some sort of guidance for the customization [124].

In the area of computer aided agile method engineering, limited studies have been conducted. Eclipse Process Framework Composer (EPFC) is the approach in which entirely defined the two well-known agile methodologies (SCRUM and XP) provide a means to enact these methodologies [21]. But the limitation of the said prototype is that it does not provide suitable support for assembly-based method engineering. Mirakhorli et al. [125] introduced a prototype box for the selection of agile methodologies which help the method engineers in classification of projects and selection of agile methodologies and practices. They provided only limited practices for project at hand and no support for assembly-based SME as well as for distributed environment.

Abad et al. [21] also developed a prototype to construct situation specific Agile Methodologies but they didn't provide adequate support for fragment definition. In addition, practitioners could not enhance or create new fragments for new situations and distributed projects. Furthermore, they adopted the System Process.

Engineering Meta Model (SPEM 2.0) is standard for method decomposition introduced by Object Management Group (OMG, 2008). At the same time, the limitations of SPEM 2.0 Meta model have been resolved by Software engineering Meta model for development methodologies SEMDM introduced by (ISO, 2014) [68].

User Satisfaction: This parameter describes the satisfaction level of the user. The situational method Engineering has been successfully generated from the ***3C-SAME*** prototype so results shows strong inclination towards user satisfaction.

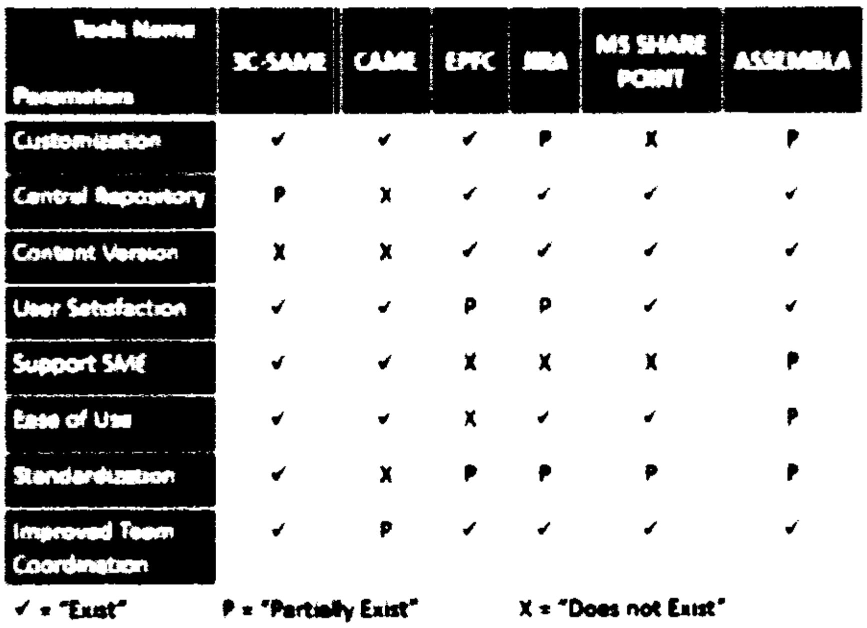
Ease of Use: This parameter is used to investigate the level to which user finds the proposed solution easy in usability. Workforce of the case study strongly agree that the suggested research solution is easy to use.

Customization: The prototype has the ability to provide customization according to the situation of project, team and organization.

Standardization: The prototype has the ability to standardize all fragments in order to formulate new method.

Content Version: This parameters deals with the changes and their impact and effect on the system.

Table 5.4: Comparison with Existing Related Studies



Tools Name / Parameters	3C-SAME	CAME	EPFC	MBA	MS SHARE POINT	ASSEMBLA
Customization	✓	✓	✓	P	X	P
Central Repository	P	X	✓	✓	✓	✓
Content Version	X	X	✓	✓	✓	✓
User Satisfaction	✓	✓	P	P	✓	✓
Support SME	✓	✓	X	X	X	P
Ease of Use	✓	✓	X	✓	✓	P
Standardization	✓	X	P	P	P	P
Improved Team Coordination	✓	P	✓	✓	✓	✓

✓ = "Exist" P = "Partially Exist" X = "Does not Exist"

Improved Team Coordination: Improved team coordination is the objective of the research. Case study results (See chapter 6) clearly indicate that team coordination improved by assigning the task.

We presented the structure of existing prototype in order to compare the effectiveness of the proposed prototype in terms of situational method composition. Table 5.4 shows comparison of the proposed prototype with existing related studies. We adopted the parameter from ISO based model for the comparison of the proposed work. The comparison table clearly shows that the existing related works are generic and their focus is towards project handling activities, whereas, our proposed work is also helpful for the software industry in order to provide the selection of suitable method according to situation specific software development.

5.8 Chapter Summary

Working prototype is a collaborative activity and software projects are normally complex and versatile in nature. Therefore, it is very necessary for GSD teams to work collaboratively to attain set goals for successful method selection in dispersed environment, during Agile GSD development. *RQ4: What are the Agile GSD situations that have been exploited for a* *solution?* In such a critical situation, when different teams are working on a task collaboratively, they are required to choose an appropriate method for software development. These Agile GSD teams are facing various challenges such as team coordination in order to customize method as per situation. Researchers have identified these challenges and continued to present solutions such as video conferencing, language choice and language translators, email, instant messaging and discussion boards. The working prototype meanwhile is intended to provide practical benefits and support for the GSD managers and Agile practitioners during method selection. These existing solutions are analyzed in order to evaluate the effectiveness of our proposed solution. It can be clearly seen in the results (See chapter 6) that the proposed solution is capable of customizing the Agile methods in order to support team coordination. We presented the overview of the proposed prototype. After this we presented the Meta model adopted for the application of prototype. In the next section we described the prototype specifications and prototype architecture. In addition, we presented the components of the ***3C-SAME*** prototype. Furthermore, we presented the few interfaces of prototype.

Chapter 6: Evaluation of the *3C-SAME* Framework

6.1 Chapter Overview

We have presented the prototype support for framework in the previous chapter, which is an important and effective way of proving framework into work. In addition, preliminary working of the prototype along with situational factors plays a vital role in improving the tailoring and customization of Agile software development practices in a global environment, which enhances, and extends the role of ***3C-SAME*** situation specific framework.

This chapter enables the GSD managers and practitioners in attaining necessary knowledge in order to evaluate the ***3C-SAME*** framework. The purpose of the evaluation is to directly address the effectiveness, in order to formulate novice methods according to situations and also to evaluate the ISO 9126-1 parameters such as time behavior about the framework.

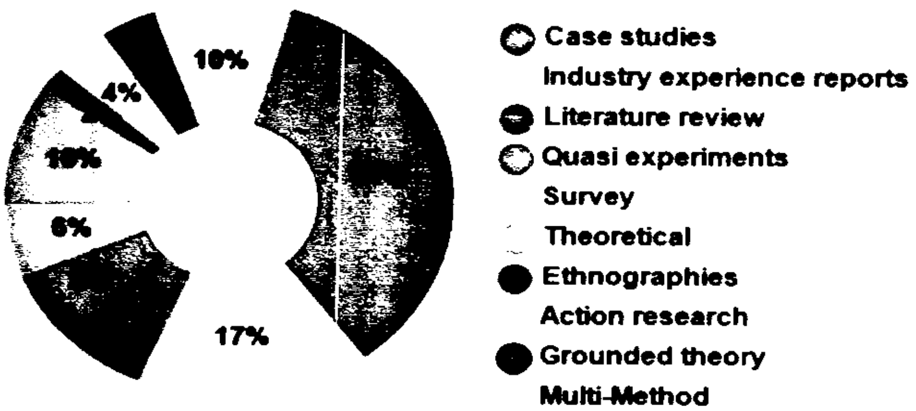


Figure 6.1: Trends of Evaluation Methods [76, 102, 104, 122, 126, 134]

It is more appropriate, to consider empirical research methods for the analysis software engineering related studies [126]. As most researchers adopt empirical methods in order to evaluate the proposed in a scientific and disciplined way so it may be applied in real software industry. These reasons compel the researcher to adopt empirical methods for analysis [128] of the most relevant research methods in software engineering such as Experimental Study, Case Study, Grounded Theory and Action Research. Figure 6.1 presents the distribution of empirical studies according to research type. The percentage of case studies (i.e., 35%) and industrial experience reports (i.e., 17%) reveal the interest of researchers and practitioners. It is vital to mention that case study methods dominate empirical software engineering research as compared to other research methods.

6.2 Evaluation Methods the *3C-SAME* Framework

We have evaluated our proposed *3C-SAME* framework with following ways.

- ISO/IEC 9126-1 Standard for Evaluation
- Evaluation Strategy: Experiment and Participant Feedback
- Evaluation using Case Study

6.3 ISO/IEC 9126-1 Standard for *3C-SAME* Framework Evaluation

We applied ISO/IEC-9126-1 Quality Model for the evaluation of the proposed *3C-SAME* framework. It is an international quality standard [108, 138], which is often used for the evaluation of software products, frameworks, models such as e-book, test specifications, ERP systems and B2B applications and ASEAME process framework. In addition, ISO 9126 model is also more appropriate to provide evaluation when multiple stakeholders are directly affected by the practical solution.

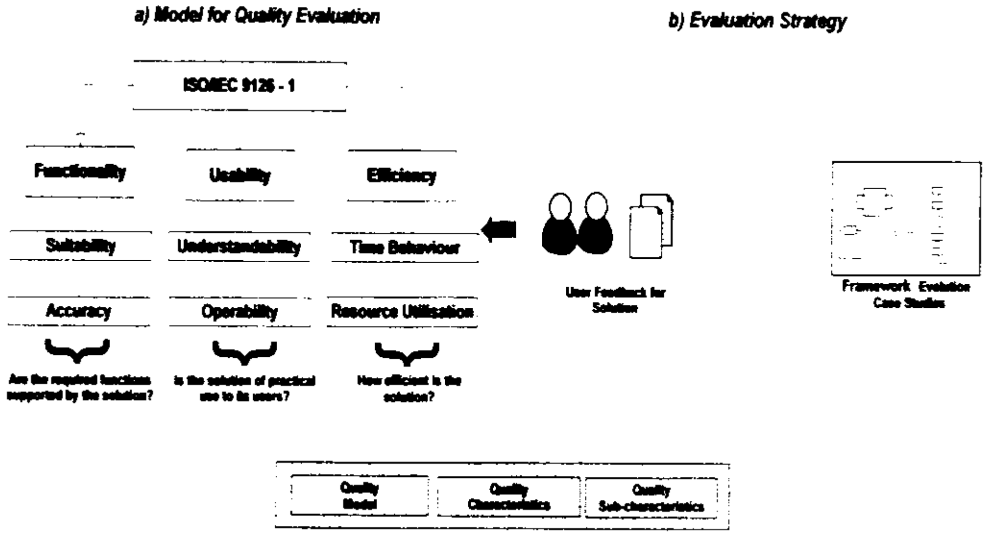

a) Model for Quality Evaluation
b) Evaluation Strategy
ISO/IEC 9126-1
Functionality
Usability
Efficiency
Suitability
Understandability
Time Behaviour
Accuracy
Operability
Resource Utilisation
Are the required functions supported by the solution?
Is the solution of practical use to its users?
How efficient is the solution?
User Feedback for Solution
Framework Evaluation Case Studies
Quality Model
Quality Characteristics
Quality Sub-characteristics

Figure 6.2: Overview of Evaluation of the *3C-SAME* Framework

Evaluation of *3C-SAME* framework components has been conducted in different ways such as experimentation, case study and ISO-9126-1.This quality model spans six quality characterizations composed of functionality, reliability, usability, efficiency, maintainability, and portability for evaluation. Furthermore, these six quality measures further sub-divide into twenty seven sub-characters. We included only three out of six parameters: functionality, usability and efficiency to evaluate the proposed framework. We choose only three due to their relevance with our framework. The remaining three have a different perspective which is beyond the scope of the current study.

6.4 Evaluation Strategy: Experiment and Participants Feedback

The evaluation of the proposed *3C-SAME* framework is concentrated on Agile GSD Practitioners perspective. The overview and parameters for evaluation are presented in Figure 6.2. In order to evaluate the *3C-SAME* framework, the following experimental study carried out emphasized design, procedure and execution parameters.

6.4.1 Design of Experimental Study:

The context of the study is to evaluate the components of the framework. Experimental planning was designed followed by the context analysis. The planning phase is composed of a set of interrelated activities such as design and formation of teams involved in daily decision making for the smooth execution of experimental study. We have selected a computer based organization in order to execute the experimental study. The organization has over fifteen years of development experience. In order to formulate the design of the study, we performed the following activities before the execution of the experimental study.

Team formation: We selected a total of seventy nine (79) participants, randomly from the organization. Among the total participants, forty five (45) practitioners fell in the control group and thirty four (34) fell in experimental group respectively and their development experience is shown in Table 6.2 respectively. In addition, the participants in the control group followed the routine development using Agile practice along with their previous knowledge. Whereas the experimental group learned both with the routine practices combined with training sessions and framework demonstrations.

Team Training: We have presented the proposed ***3C-SAME*** framework and have also demonstrated the working prototype to the participants of the experimental group for a short period of (maximum 30 minutes a day) three days.

Instrumentation: A questionnaire was formulated [40] (See Appendix A) in order to evaluate the truthfulness or falsification of the proposed framework. In addition, the questionnaires were circulated among the participants of both control and experimental groups. This questionnaire serves as an instrument for feedback collection from the participants.

6.4.2 Experimental Procedure

Roles and responsibilities were assigned to the participants as per their organizational hierarchy. The project manager maintains the profiles of every individual participant. Allocated tasks, responses made against the allocated tasks, interaction time with proposed work and situational contents for participants were a few items of the participant profile. Keeping in view the experimental context, all participants had at least sixteen years of education in computer science and had basic knowledge of Agile methods.

The time line for the experiment was sixty days. The participants were cooperative due to the involvement of their top level management. The instrument was divided into two sections, the familiar and the unseen. The familiar section contained questions about Agile practices that were deemed to be present in the knowledge of the participants. Participants mainly gained such knowledge about Agile practices from available articles and textbooks. The unseen section participants faced difficulty in understanding Agile terms. A project manager remained within experimental place in order to motivate and provide support if required, such as reading the question statement and understanding the Agile GSD terms. The pre and post experiments results were noted on paper worksheets (See Appendix C-F) in order to calculate the result for control and experimental groups.

Table 6.1: Null Hypothesis

	Hypothesis Statement
H01	There is no significant difference between pre-test mean scores of control and experimental groups
H02	There is no significant difference between post-test mean scores of control and experimental groups.

Furthermore, hypothesis evaluation was conducted in order to investigate the cause and its possible effect on the relationship between the samples that have been used for the experimental study. We constructed three null hypothesis is shown in Table 6.1 to evaluate the sample of the current study

6.4.3 Experimental Study Execution

The scope of the study is therefore limited to pre-test evaluation and post-test evaluation and its detailed description is given below.

Pre-test Analysis of Experimental Study: The objective of the pre-test taken before any training was to evaluate the effectiveness of the treatment [136], whereas the post-test is executed after training participants. In addition, the difference between the results of these two tests was calculated. The following activities were carried out during pre-test analysis of the experimental study as shown in the Table 6.2.

Table 6.2: Pre-test Analysis Descriptive of Experimental Study

	Control Group (A)				**Experimental Group (B)**			
Experience	N		**Pre- Test Mean Score**		N		**Pre-test Mean Score**	
	45	**Sum of Mean Score (%)**	**Familiar Max=75**	**Unseen Max=60**	**34**	**Sum of Mean Score (%)**	**Familiar Max=75**	**Unseen Max=60**
±2 Years	16	3.14 (2.32%)	1.58	1.56	8	3.08 (2.28%)	1.55	1.53
± 3Years	12	3.21 (2.37%)	1.62	1.59	7	3.13 (2.31%)	1.57	1.56
±4 Years	7	3.34 (2.47%)	1.70	1.64	8	3.22 (2.38%)	1.62	1.60
±5 Years	4	3.53 (2.61%)	1.78	1.75	6	3.31 (2.45%)	1.67	1.64
± 6 Years	6	3.66 (2.71%)	1.84	1.82	5	3.39 (2.51%)	1.72	1.67
Mean			3.66		Mean		3.23	
Std. Deviation			0.22		Std. Deviation		0.12	

The Pre-test score for control and experimental groups is shown in Table 6.2 which evaluates the performance of participants in the control and experimental environment. The experiment executed showed improvement in the usability benchmark. We have selected two groups, one control and the other experimental, for experimental study from organizations.

The aim was to evaluate the expertise level of the two groups. In this context, when results were presented in form of statement, hypothesis is applied in order to judge the logical relationship between two are more variables [136] as Hypothesis H01 outcome may be true or false.

In order to evaluate the hypothesis (H01), we performed t-Test to evaluate the results of the significant difference between two groups as shown in the Table 6.3. The t-Test compared the results of each group and then it determined the statistical significance between the groups. Mostly statisticians recommended that the acceptance and refusal of the hypothesis is established on the P-value results [136, 139]. The recommended P-value is 0.05 and in our case after the evaluation of the 3C-SAME approach the p value is 0.239, which is higher than 0.05 as shown in the Table 6.3.

Table 6.3: t-Test: Two-sample Assuming Unequal Variance (Pre-test)

	Pre-test Scores
t-test	0.712
Observation	79
Mean Difference	0.694
p-Value	0.239

The hypothesis (H01) was accepted based on the analysis of the significant value. Therefore, it has been proved that there is no significant difference between the results of both groups.

Post-test Analysis of Experimental Study: The overriding objective of the post-test is to find the statistical significance between control and experimental group after applying specific treatment [137, 140]. We apply quality parameter accuracy in order to evaluate the proposed approach. The Table 6.4 shows the post result score of the control and experimental group. Most of the results clearly show an increase in the post-test scores (See Appendix E-F), especially in the experimental group.

Table 6.4: Post-test Analysis Descriptive of Experimental Study

	Control Group (A)				Experimental Group (B)			
Experience	N		Post- Test Mean Score		N		Post-Test Mean Score	
	45	Familiar Max=75	Unseen Max= 60	Sum of Mean Score (%)	34	Familiar Max=75	Unseen Max=60	Sum of Mean Score (%)
±2 Years	16	2.49	2.47	4.96 (3.67%)	8	3.89	3.5	7.39 (5.47%)
±3 Years	12	2.50	2.48	4.98 (3.69%)	7	3.96	3.54	7.50 (5.56%)
±4 Years	7	2.51	2.49	5.0 (5.70%)	8	4.01	3.43	7.44 (5.51%)
±5 Years	4	2.52	2.50	5.02 (3.72%)	6	4.14	3.85	7.99 (5.92%)
±6 Years	6	2.54	2.53	5.07 (3.76%)	5	4.25	3.98	8.24 (6.10%)
Mean			5.07		Mean		7.71	
Std. Deviation			0.04		Std. Deviation		0.38	

Table 6.4 shows two types of contents which are unseen contents and familiar contents. Familiar contents are those contents that participants have gained training before the execution of experiments such as fragment management, whereas unseen content is not seen before the start of the experiment. The familiar content value increases as compared to unseen content. The reason of this increase in number is due to content training which makes execution for participants easier.

Table 6.5: General Comparison of the Results

Groups	**N**	**Pre Test**		**Post- test**	
		Mean	**Std. Dev**	**Mean**	**Std. Dev**
Control Group	45	3.66	0.22	5.07	0.04
Experimental Group	34	3.23	0.12	7.71	0.38

The mean of the control group value is 3.66, standard deviation value is 0.22 whereas the mean of experimental group value is 3.23 and standard deviation is 0.12.The difference between these two results, clearly indicate that framework based approach is easy to use and easy to operate, therefore training creates the difference in results.

We have presented the comparison of results between of post-test and pre-test results in the Table 6.5, in order to judge the effectiveness of our proposed approach. The results of the both groups show improvement in results for both familiar and unseen items with the experimental group showing a larger improvement in value on both pre-test and post-tests. In addition, overall statistical tests were used in order to get in depth evaluation findings.

Table 6.6: t-Test: Two Sample Assuming Unequal Variance (Post Tests)

	Post-test Scores
t	17.26
Observation	79
Mean Difference	30.18
p-Value	0.000

Furthermore, the pre-test evaluation was performed in order to evaluate the Hypothesis *(H02)* usability of practitioners in both groups after applying the treatment.

Table 6.6 clearly shows the results of t-Test which is used for analysis of the hypothesis (H02) in order to find the variance between experimental groups. Table 6.6 has shown the t-Test score between control and experimental groups. The value in row (p-Value.) is less than alpha value 0.05 which clearly reflects the learnability between the results of both groups has considerably changed so the Hypothesis H02 is rejected.

Table 6.7: Conclusion of the Hypothesis Test

	Hypothesis Statement	***p*-Value**	**Decision**
H01	There is no significant difference between pre-test mean scores of control and experimental groups	0.239	Accepted
H02	There is no significant difference between post-test mean scores of control and experimental groups.	0.000	Rejected

Table 6.9 provides a conclusion of the hypothesis tests. The hypothesis (H01) is accepted due to p-Value. H02 are rejected as findings indicate that there is a substantial change in scores of participant in both groups after the treatment was applied.

The primary goal of this experiment is to assess the effectiveness of the situation specific framework based approach in order to support distributed agile activities and provide customization for situational methods in a globally distributed settings. For the evaluation of the *3C-SAME* Framework, it was made available to the practitioners.

Therefore, the evaluated tasks of the experimental groups involved the management features such as creating a new situational method, a new team kind, assigning the team to the fragment, creating a new participant, adding a participant to a team, defining a task kind of a project and so on.

6.4.4 Limitations of the Experimental Study

The limitations of experimental study is given below:

The experiment results has been mainly positive even though there are some issues that need to be addressed in a possible future work. The first finding to clarify is that this *3C-SAME* Framework has been designed in order to support the practitioners as a prototype; therefore, it may not *suitable* for a real world scenario. Taking into account this premise, all participants have found this kind of application useful, since it provides many advantages in comparison with the existing related approaches such as Jira. One participant commented: "It is a standalone approach that can make distributed teams to coordinate in an efficient manner without any overhead".

Another participant also provided an explanation of the *3C-SAME* framework compared with the traditional development models such as Jira. The *3C-SAME* framework also provides a fast and easy approach for accessing a situational method: "You can quickly get situational method from the *3C-SAME* framework based approach in comparison to the traditional approaches such as water fall, which normally requires significantly more time to reach it" and "... in few situations such as Lean development, the current study may have the problem of accessibility and customizations etc." because it is only limited to XP and Scrum method.

As claimed by the authors of the *3C-SAME* approach, the method situation specification is one of the most important artifacts of the *3C-SAME* approach, which helps to enhance awareness of the situational factors among practitioners. Most of the participants agreed with how the *3C-SAME* approach is presenting the technique kind information such as XP Practices. However, one of the participants also provided a suggestion for making the producer kind more attractive to users. As suggested by this participant, the presentation of the fragment management may be a good point for such type of approach. In general, information must be customizable wherever it is displayed."

Another limitation of the current study is availability of the comprehensive knowledge and expertise of Agile as well GSD participants.

6.5 Evaluation using Case Study

In this section, we have presented case studies as a research method and provide reasons for the selection of the case study [120, 126]. We collected research data mainly concentrating on empirical evaluation of Agile GSD practices to select appropriate situational factors. We adopted a case study as a research method for the empirical analysis of the ***3C-SAME*** Framework. In the case study, the real-life phenomenon can be investigated in the natural settings and it may not be investigated properly in hypothetical situations. When any phenomenon is investigated in a real environment, it is known as case study [122,128].

The case study is also applicable in other disciplines as well, such as medicine, sociology, education, psychology, business, and law. Kitchenham et al. [135] described the key benefit of case study usage i.e. real time results achieved in real situations. In addition, he presented the few disadvantages of case studies, such as there is no guarantee about the consistency in results which may change as the situation changes [99].

6.5.1 Demographic Information

For the evaluation of our proposed ***3C-SAME*** framework, we have generated a situation specific agile methodology for a real-world software project. It is an exploratory case study project. We have selected a software development organization working in healthcare domain since the last 15 years. Due to an organizational privacy policy, we will not mention the name of the organization. The site office is situated in Rawalpindi, Pakistan and Dublin, Ireland. All the development related activities are performed in the Pakistan office. Ireland office is responsible for marketing, managing business and fulfilling requirements of clients who mostly belong to Ireland. The organization has more than 300 employees across the globe that seek to employ new technologies to improve product performance.

The nature of the project in this case is in order to achieve the optimal solution using home care provider organizations. It was an ongoing project and the organization was working on it since 2014. A semi agile methodology was already in place as a development methodology of the project that included a software team having basic knowledge about agile methodologies and the domain knowledge of home care providers. Product backlog was managed in Team Foundation Server and new requirements were gathered through emails and Skype meetings. The new release update was being pushed after 3 months at the end of Iteration. So there were 4 iterations in a year to deliver new features to the clients. Deadlines were not being met and clients were also unhappy. Developers worked extra hours to meet the deadline. The development team comprised of eleven developers (including both application and database developers) and two quality assurance engineers. The team has a mix of seniors and juniors and team members have diverse skills of technologies being used for the development of applications. There was also a team of 5 analysts, composed of requirement analysts, solution specialists and requirement engineers who were responsible for eliciting the requirements from customers. There were six customer organizations which have diverse requirements according to their organizational workflow and many requirements in common because each organization was working in the same domain.

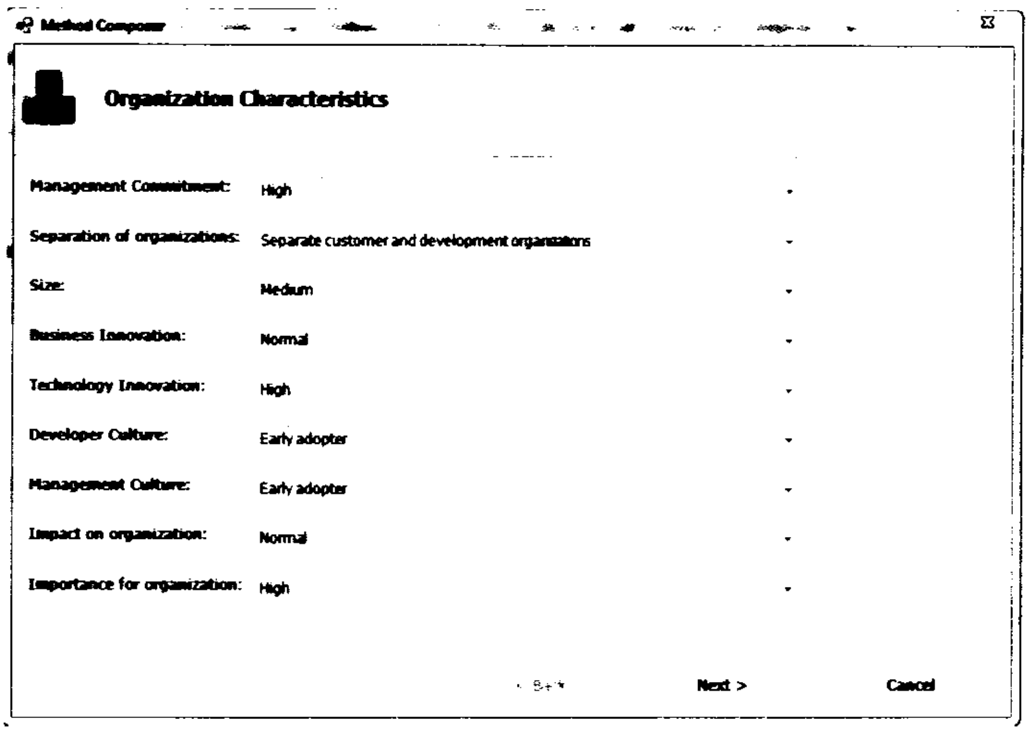
6.5.2 Case Study Execution Procedures

The case study is executed with a brief introduction of agile values and principles. In addition, two well-known agile methodologies, Extreme Programming and Scrum were parented. The Prototype was demonstrated by using the agile methodologies scenarios. Only development teams attended the session as our objective is to explore the situation specific activities. This session was conducted in a conference room in the Rawalpindi office. This session started in the morning and finished in 3 hours.

Then we organized another session in the afternoon at the same location to analyze the context of case study [141]. Firstly we gathered the system specification for execution and for the evaluation of the proposed approach. In addition, these specifications were used as inputs in order to evaluate the results of the proposed approach.

6.5.3 Evaluation Results

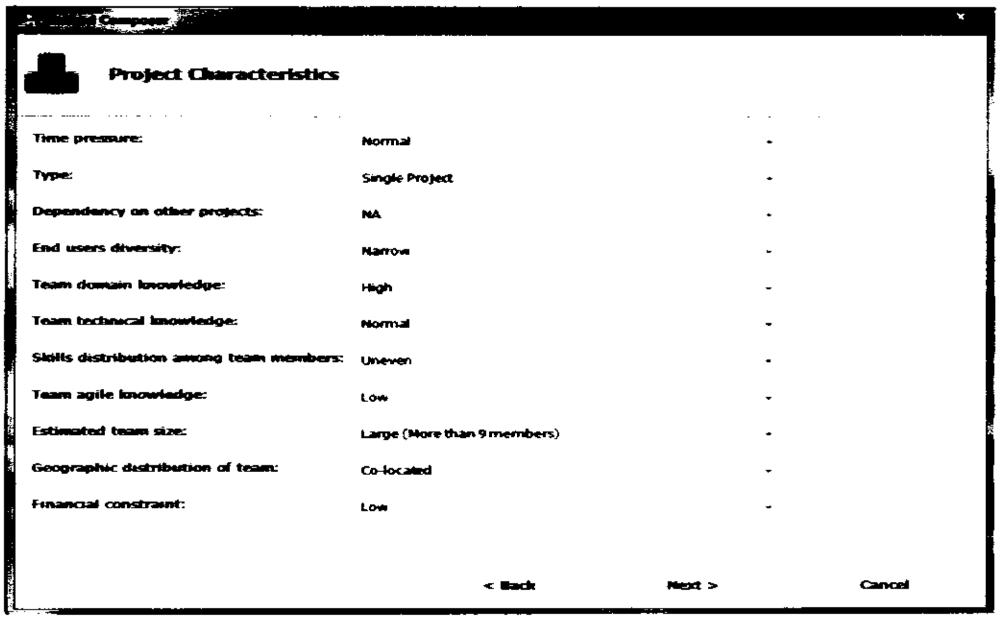
We adopted Organization characteristics, as contextual factors, along with other possible values. During the second session the team was able to extract methods for data gathering of the results which are presented in Appendix C-F.



Management Commitment:	High
Separation of organizations:	Separate customer and development organizations
Size:	Medium
Business Innovation:	Normal
Technology Innovation:	High
Developer Culture:	Early adopter
Management Culture:	Early adopter
Impact on organization:	Normal
Importance for organization:	High

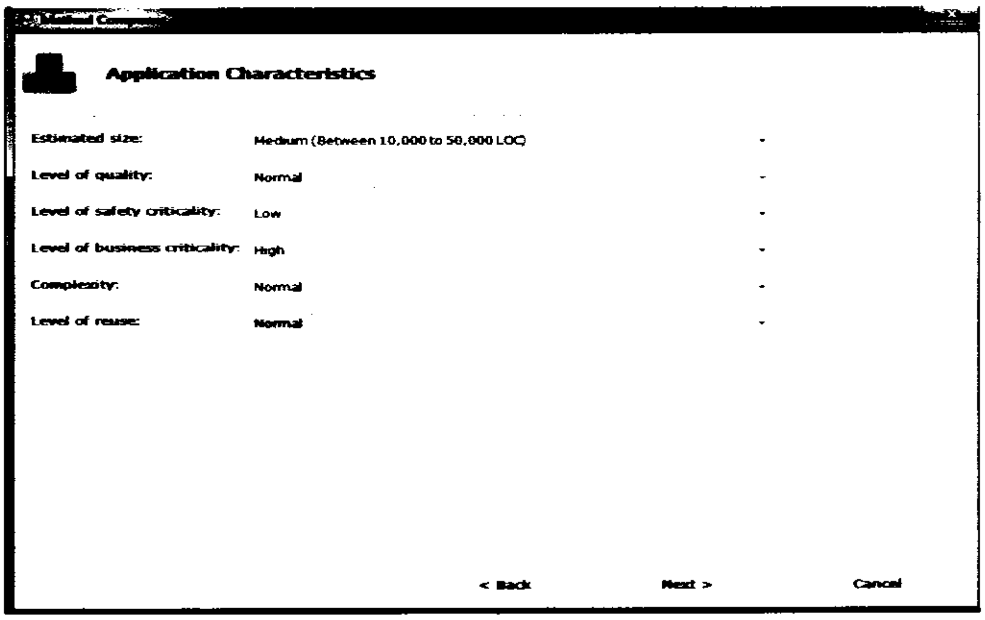
Figure 6.3: Organization Level Characteristics

The extracted situational characteristics were then given as an input to the Method Composer of the Approach to generate the situation specific methodology for the next iteration of the project under investigation. Figures 6.3, 6.4 and 6.5 show the method composer screens with input to represent organization, project and application characteristics respectively.



Time pressure:	Normal
Type:	Single Project
Dependency on other projects:	NA
End users diversity:	Narrow
Team domain knowledge:	High
Team technical knowledge:	Normal
Skills distribution among team members:	Uneven
Team agile knowledge:	Low
Estimated team size:	Large (More than 9 members)
Geographic distribution of team:	Co-located
Financial constraint:	Low

Figure 6.4: Project Level Characteristics

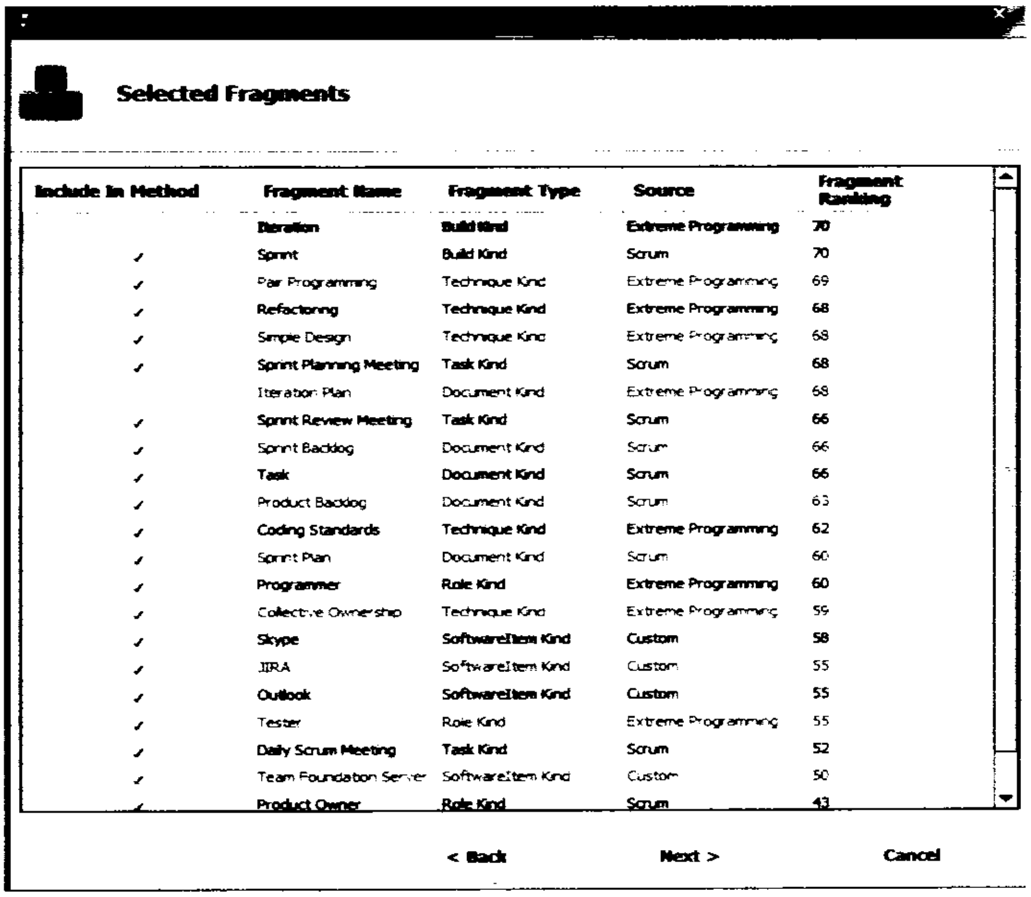


Estimated size:	Medium (Between 10,000 to 50,000 LOC)
Level of quality:	Normal
Level of safety criticality:	Low
Level of business criticality:	High
Complexity:	Normal
Level of reuse:	Normal

Figure 6.5: Application Level Characteristics

6.6 Generated Agile Methodology

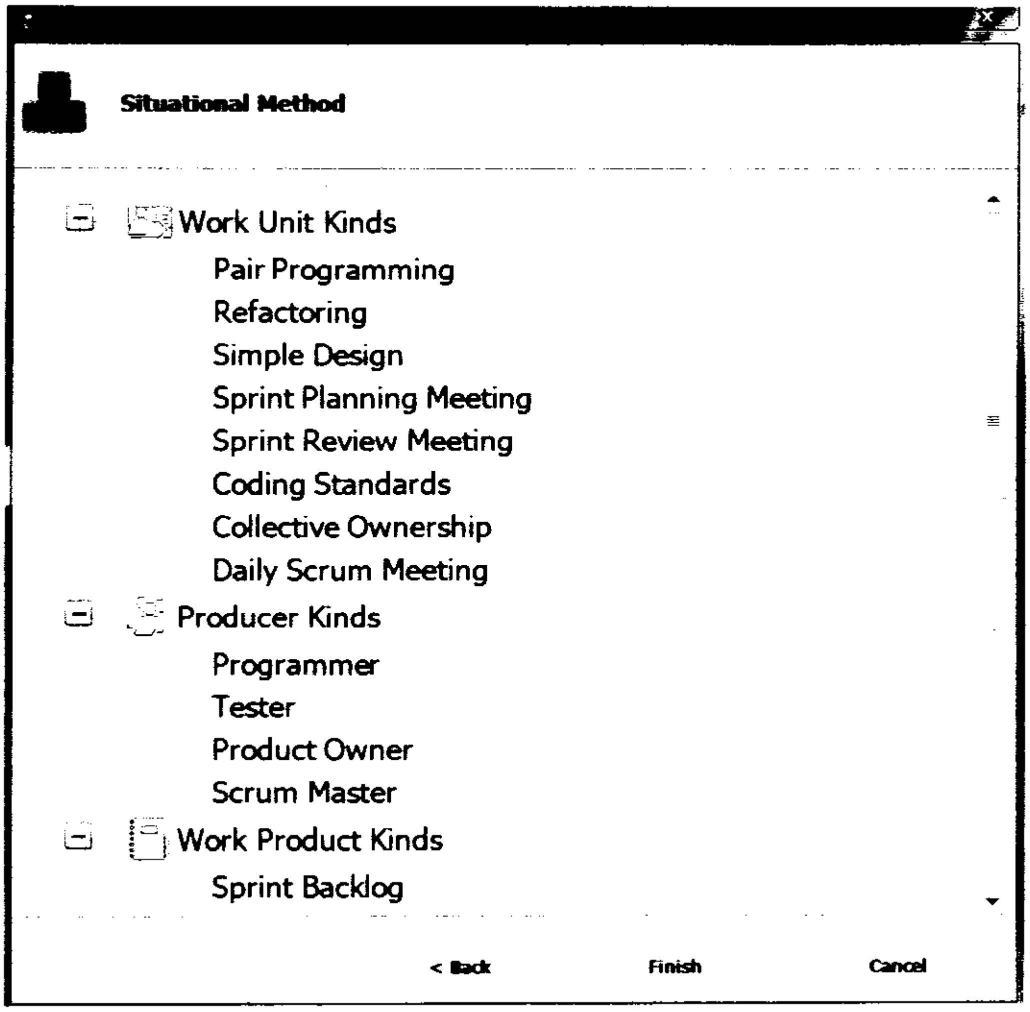
Based on situational characteristics, the Prototype has generated the agile methodology as shown in Figure 6.6 and Figure 6.7. The generated methodology was a mix of extreme programming and scrum. For the technical aspects of the project, the Prototype mostly suggested extreme programming fragments and for management it provided scrum fragments. Due to similarity between Iteration and sprint fragments, we have not included Iteration fragment in the final methodology. The methodology could be considered as a lightweight methodology. The development team followed that methodology over a full sprint to get the insight on its suitability for the project. Sprint duration was a two month period to deliver the key features requested by different clients.



Include In Method	Fragment Name	Fragment Type	Source	Fragment Ranking
	Iteration	Build Kind	Extreme Programming	70
✓	Sprint	Build Kind	Scrum	70
✓	Pair Programming	Technique Kind	Extreme Programming	69
✓	Refactoring	Technique Kind	Extreme Programming	68
✓	Simple Design	Technique Kind	Extreme Programming	68
✓	Sprint Planning Meeting	Task Kind	Scrum	68
	Iteration Plan	Document Kind	Extreme Programming	68
✓	Sprint Review Meeting	Task Kind	Scrum	66
✓	Sprint Backlog	Document Kind	Scrum	66
✓	Task	Document Kind	Scrum	66
✓	Product Backlog	Document Kind	Scrum	63
✓	Coding Standards	Technique Kind	Extreme Programming	62
✓	Sprint Plan	Document Kind	Scrum	60
✓	Programmer	Role Kind	Extreme Programming	60
✓	Collective Ownership	Technique Kind	Extreme Programming	59
✓	Skype	SoftwareItem Kind	Custom	58
✓	JIRA	SoftwareItem Kind	Custom	55
✓	Outlook	SoftwareItem Kind	Custom	55
✓	Tester	Role Kind	Extreme Programming	55
✓	Daily Scrum Meeting	Task Kind	Scrum	52
✓	Team Foundation Server	SoftwareItem Kind	Custom	50
✓	Product Owner	Role Kind	Scrum	43

Figure 6.6: Selected Fragments for Case Study in Action

The Prototype was installed on each team member computer and connected to a centralized methodbase. The team used the Prototype whenever they needed an understanding of the fragments and their relationship with other fragments to ensure its working. Then method composer generates the situational method.


Situational Method

- Work Unit Kinds
  - Pair Programming
  - Refactoring
  - Simple Design
  - Sprint Planning Meeting
  - Sprint Review Meeting
  - Coding Standards
  - Collective Ownership
  - Daily Scrum Meeting
- Producer Kinds
  - Programmer
  - Tester
  - Product Owner
  - Scrum Master
- Work Product Kinds
  - Sprint Backlog

Figure 6.7: Situational Agile Method as Outcome of Case Study in Action

6.7 Case Study Lesson Learned and Limitations

The feedback of case study has been mainly positive and after successful execution of the case study we find some lessons learned which is presented below.

One of the features of the ***3C-SAME*** approach that motivates the participant's interest is its compatibility to ISO/IEC 24744. It helps to enhance the applicability of creating existing contextual factors, such as size etc so method engineers did not have to create a new method again and again. This feature helps the method engineer and GSD managers to save time in managing the situation specific method. Additionally, the feature for displaying different project characteristics and presenting available fragments in collaborative sites also helps to enhance team coordination among team members. One of the participants who is a participant of the *3C-SAME* approach stated, "most of the developmental features of the *3C-SAME* approach are helpful". However, since the *3C-SAME* approach is at the initiation of the development process, there are still many features that need to be incorporated in the approach to make it more attractive to users for example one of the participants suggested potential features for the *3C-SAME* approach: ".. Features like task sharing can be added to the Agile software development" and another participant also suggested that "... in the project details page there must be a field where users can attach a text document". In the near future, we are going to analyze these requirements carefully to verify whether it is reasonable and feasible to incorporate these features in the *3C-SAME* approach. One of the issues of the *3C-SAME* approach was the navigation mechanism. At the current stage the *3C-SAME* approach did not implement a flow between phases of the *3C-SAME* approach (e.g., fragment management, method management, administration). Also, there was a lack of a mechanism for preventing users navigating from one phase to another if the current phase was not completed.

This may cause the *3C-SAME* approach to present an empty page when there is no data available. For example, one participant complained: "... nothing is displayed in the communication tab... the visualization of connectivity map of some projects lead to server error". This mistake could be explained by that the participant was logged in to the *3C-SAME* approach as an administrative user. He/she created a new project but he/she did not assign any team for the project yet. He/she then clicked on the role kind tab for viewing status of a different role kind between team members. As a result, there were no available team members to show their role kind status. A further improvement of this can be to incorporate a mechanism for preventing users from using the functionalities that are not available at a certain phase.

The following limitations have been founded with the case study conducted.
The main limitation concerns the availability of Agile and GSD experts Participants. This means that the design of the tasks and the questionnaire has been thought out according to the number of available participants. Moreover, the *3C-SAME* approach is not compatible with the cloud infrastructure. The current study handled only three GSD situations such as Application, Project and Organization.

One limitation of the statistic analysis is sample size. Limited participants participated in interview process and gave their responses as per rating scale.

6.8 Summary of Chapter and Discussion over the evaluation of *3C-SAME* framework based Approach

The evaluation of the framework is presented in this chapter to evaluate the effectiveness of the *3C-SAME* approach. *RQ 5: What evaluation methods have been used to validate the proposed solution?* We have applied three renowned research methods such as a case study, experimentation and ISO based model. The detail specification along with software architecture of *3C-SAME* prototype is presented in the previous chapter. Empirical analysis shows clear evidence for the notion that Agile GSD practices apply SME principles to formulate situational methods. Practitioners/researchers report for these reasons that organizations are moving from traditional development approaches to Agile based approach as it has the ability to deliver products closer to customer's expectations. It is vital to recognize the overall integration of the two popular trends in order to benefit from combining and formulating new Agile GSD according to the situation.

However, the Situational Method Engineering (SME) is a solution for the integration of more than one method and formulation of situational method. It also arises as a solution when we want to combine two approaches for the formulation of a new paradigm. Hence SME aims at providing method construction techniques by reusing existing methods, practices, techniques and approaches.

We have devised a meta model along with Agile GSD practices which is divided into multiple phases including method composition and enactment. In proposing a solution framework, roles, artifacts, and activities are clearly defined and elaborated further to explain in order to formulate a situational development method.

For the evaluation process, we have generated success factors from Figure 6.2. On the basis of these parameters we have evaluated *3C-SAME* approach using existing three evaluation methods as shown in Figure 6.8. The previous section shows the results of experimental study for the effectiveness of Agile GSD practices to formulate a situational method. There were two treatments where a control group and an experimental group was assigned to teams in a development context. It is evident from the analysis of the results that the satisfaction level of the participants/subjects is positive towards ***3C-SAME*** Approach.

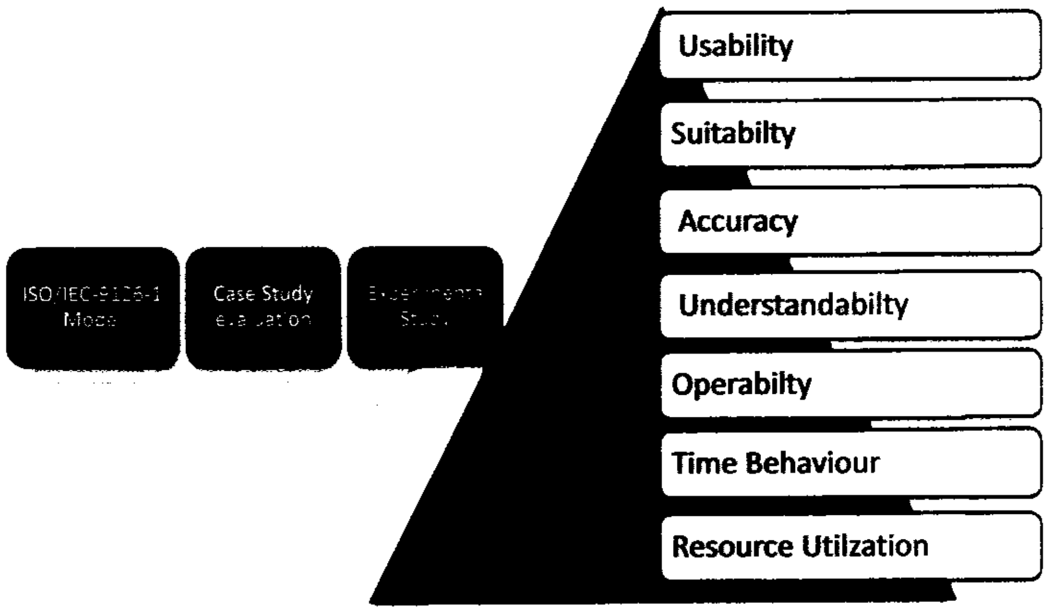


Figure 6.8: Summary of Evaluation

In experiments, teams were evaluated and tested on the basis of two treatments presented in Table 6.2 and 6.3. In the experimental group, the team applies the appropriate Agile practices along with SME principles. Table 6.3 indicates the good amount of satisfaction of participants with this treatment. Majority of participants' results shows positive feedback in terms of effectiveness of Agile practices to formulate situational method.

We performed evaluations of the ***3C-SAME*** framework through case studies and we generated a situation specific agile methodology for a real-world software project which is exploratory in nature. We selected a software development organization working in Agile GSD domain for 15 years. Their results clearly indicate that the team applied the appropriate agile practices, but did not incorporate the strategies for SME principles with project development. The disclosed result after analysis did not indicate a greater amount of satisfaction of participants with this second approach in which they adopted the strategies to overcome the SME rules as compared to traditional development approaches.

Table 6.8: Summary of Evaluation Results with ISO/IEC 9126

Characterization	Sub-Characterization	Support in 3C-SAME	Semantics
Functionality	Suitability	✓	The agile fragments are suitable up to some extent for the formulation of new methods.
	Accuracy	P	Their situational methods are accurate with respect to agile fragments.
Usability	Understandability	✓	The 3C-SAME approach is provided the facility of ease of use.
	Operability	✓	3C-SAME supports method development notations.
Efficiency	Time Behavior	P	The 3C-SAME approach reduces the time of GSD manager for method customization.
	Resource Utilization	P	There is adequate support for resource utilization.
✓ = "Exist"	P = "Partially Exist"		

We apply our prototype to assess the effectiveness of the Approach for supporting situational method. For the evaluation of the ***3C-SAME*** Approach, it was made available to the practitioners. We managed to seek case study results for this evaluation. Their feedback was generally positive even though there were some issues that needed to be addressed. It is vital to mention that even a single participant did not disagree with the effectiveness of agile practices. The second group did not apply the appropriate Agile practices but applied SME principles. The outcome did not specify a considerable amount of satisfaction of participants with this evaluation. The summary of the feedback from the project manager is presented in Table 6.8.

It is concluded that agile practices along with SME strategies can be effectively used in distributed environments to formulate situational methods. Analysis of results after executing evaluation methods shows that the ***3C-SAME is*** an effective approach as compared to traditional approaches.

Chapter 7: Conclusion and Future Work

7.1 Chapter Overview

We have presented the evaluation methods in order to evaluate the effectiveness of the framework in the previous chapter. We evaluated our proposed framework through the ISO/IEC-9126-1 quality model, case study and experimental study.
This chapter presents the implications of the proposed framework, research findings, and the novelty of the research and future research directions of the current study. In addition, the overriding objective of this chapter is to map the research objective with research outcomes. The increasing and complex project situations need to employ resources on a global scale that has resulted commonly known as Global Software Development (GSD). In addition, GSD teams work round the clock to fulfill the complex and competitive market situation to optimize and decrease costs by utilizing time zones effectively.

However, apart from the advantages in GSD, it also entails certain challenges due to distance and time difference such as lack of control over the team coordination that leads to project failure. With the passage of time, the complexity of software has increased greatly. It is for these reasons that organizations are moving from traditional development approaches to Agile based approach as it has the ability to deliver products closer to customer expectations. This research is to design the structure of the ***3C-SAME*** Framework in order to formulate situational methods to help Agile GSD practitioners to provide a productive solution.

7.2 Summary of Research Contributions

We have presented the key finding of the current study and also provide mapping with the previously identified research objectives (See Chapter 1). Keeping in view the research objective, numerous research contributions have been concisely given below:

Objective 1: The main objective is to critically analyze prevailing state of the art in Agile GSD, including the existing taxonomies, approaches, models and frameworks.

Contributions: We reviewed the existing related available literature (See Chapter 2) about global software engineering, Agile GSD software development, Agile GSD situational method engineering, existing taxonomies, frameworks, approaches, and models. We presented the steps of mapping study, conducting mapping process and data extraction strategy. We also described the global software development challenges, existing Methodologies to mitigate these challenges, outcome of literature review and broader overview of the proposed solution.

Objective 2: Investigate the available challenges in GSD as reported in the literature to formulate an appropriate situational taxonomy to classify the possible GSD challenges in a hierarchical order.

Contributions: We proposed a *3C* taxonomy (See Chapter 3) for global software development teams. We presented an overview of GSD challenges, analysis of existing taxonomies, the process of creating a taxonomy, *3C* Situational taxonomy creation, taxonomy evaluation, *3C* taxonomy value addition and discussion over practical implications.

Objective 3: Design a Situation Specific Framework based approach to support practitioners to formulate situational methods for Agile GSD development.

Contributions: We have presented the proposed situation specific framework (See Chapter 4). In addition, this generic model is also designed in order to develop the Agile GSD projects. Furthermore, the framework is mainly focused on situational method construction, method enactment and Meta model adoption in order to formulate rules, activities and artifacts by applying SME. The main purpose of this framework is to provide an innovative and enriched approach for practitioners. Furthermore, we compared our proposed approach to existing related work, role of Meta model and *3C-SAME* framework components and its capabilities.

Objective 4: To evaluate the situational framework in terms of improvements to usability and effectiveness.

Contributions: We have characterized the specification of the *3C-SAME* framework, as a prototype application (See Chapter 5) for the validation of situational methods. The point is to draw the fact that prototype support for framework is an important and effective way of proving the framework theory into practice, which is more practical and easier to use for software industry practitioners. We evaluated the *3C-SAME* framework to directly address the effectiveness, in order to formulate situational methods according to situations and also to evaluate the ISO 9126-1 parameters (See Chapter 6) such as time behavior about the framework. In addition to case study, experimental study is also used for evaluating the framework.

7.3 Novelty of the Research:

It is vital to recognize the overall integration of the two popular trends in order to benefit from combining Agile methods with GSD. Despite their significance, representations are not the only success factor in software development process. It is generally acknowledged that it is not feasible to have a universal method without any modifications used in all situations. The Situational Method Engineering (SME) matures as a solution in order to formulate a new situational method. It also arises as a solution when we want to combine two approaches for formulating a new paradigm. Hence SME aims at providing method construction techniques by reusing existing methods, practices, techniques and approaches. The newly developed methods are more flexible and better adapted according to the distributed situations.

The *3C-SAME* framework based approach is effective to handle situations is an agile methodology under distributed development to fulfill the gap between research and practice. In addition, the uniqueness of the *3C-SAME* framework is that it is based on situational taxonomy and basic generic model while other similar approaches such Emam et al. Framework [49] is only based on literature review.

Furthermore, we evaluate the *3C-SAME* framework with three methods: ISO based model, experimental study and case study, whereas Emam are all. [49] Evaluated by case study only and did not even apply SME to formulate the framework.

7.4 Future Research Dimensions

The current research may be applied and further extended in different dimensions in the future, as given below:

1. Although current research has presented a vital contribution such as proving conceptual work framework into practice, in future the ***3C-SAME*** framework can be extended and further refined to include cloud computing [57, 58] as a service to support knowledge repository.

2. We have adopted the term Fragment for our proposed framework in accordance with SEMDM and more specifically Agile Fragment. The main sources of agile fragments are existing agile methodologies like Scrum and XP. These fragments are extracted from their primary sources as per standard format. In the future, current research can add Lean fragment as well, which is another popular agile methodology.

3. The current research can strengthen to support artificial intelligence techniques such as Case Based Reasoning (CBR) [70], which is applied to reuse the existing knowledge repository and also enhance the role experience factory.

4. The application of the framework is based on concepts and their associations. In our context, we have preferred ISO/IEC-24744-SEMDM Meta model, in order to formulate a framework for the tailoring and customization of Agile GSD methodologies. In future Meta model can be designed to formulate new approaches.

List of Author's Publications from the Thesis

1. Yaser Hafeez, Sohail Asghar, Aakash Ahmad and Ayaz Hussain (2016), "A Taxonomy to Support Team Coordination in Global Software Engineering Environment" Published in Journal of Computational and Theoretical Nanoscience, Vol. 13, No. 5, PP: 3238–3247, 2016, ISSN: 15461955/2016/13/3238/010, Thomson ISI Impact Factor: 1.34, doi:10.1166/jctn.2016.4981, Thomson Reuters Master Journal List.

2. Ambar Sarwar, Yaser Hafeez, Sohail Asghar, Khushbakhat war and Zartasha Gul (2015). "Outsourcing Agile Software Development Model Using Grounded Theory" Published in Journal of Applied Environmental and Biological Sciences (JAEBS) (Thomson Reuters ISI indexed), 2015, J. Appl. Environ. Biol. Sci., Volume: 5(12)1-1, 2015, ISSN: 2090-4274, Thomson Reuters Master Journal List.

3. A. Sarwar,k.,Bhakat & Y. Hafeez Motla "An A Situational Requirement Engineering Model for an Agile Process Ontological Framework For Requirement Change Management In Distributed Environment " Published in The Bahria University Journal of information and communication Technologies ISSN 0 0 2 9 - 5 6 9 8" Volume 51, No. 2 (2015) PP: 291- 301 April 30, 2014.

4. S. Sultana, Y.Hafeez Motla, S. Asghar, M. Jamal and R. Azad "A Hybrid Model by Integrating Agile Practices for Pakistani Software Industry "Accepted in 24th International Conference on Electronics, Communications and Computers", CONIELECOMP ISBN: 978-1-4799-3469-0/14, 2014 Cholula, Puebla, MEXICO.

5. Yaser Hafeez, A batool, Sohail Asghar,M Jamal "Role of Software Requirements to improve the Quality of Scrum Framework" Science International-Lahore ISI-indexed Journal, 26(1), 2014, ISSN# 1013-5316, CODEN: SINTE 8.

List of Author's Publications from the Software Engineering Domain

1. M., Ahsan, Y.Hafeez Motla and M., W., Azeem (2015). "An Ontology-based Approach for Handling the Issues in Requirement Engineering", Published in Proceedings of the Pakistan Academy of Sciences, Vol: 52(3), Issue September, 2015: PP: 187-200 (2015) ISSN: 0377-2969.
2. Syeda A.A., Y.Hafeez, S. Asghar, M.S. Hassan, and B. Hamid (2014). "Towards a Framework for Scrum Handover Process "Published in Proceedings of the Pakistan Academy of Sciences, Vol: 51, Issue December, 2014: PP: 265-275,(2014) ISSN: 0377-2969.
3. Bibi, S., Hafeez, Y., Hassan, M.S., Gul, Z., Pervez, H., Ahmed, I. and Mazhar, S. (2014) Requirement Change Management in Global Software Environment Using Cloud Computing. *Journal of Software Engineering and Applications*, Vol: 7, PP: 694-699. http://dx.doi.org/10.4236/jsea.2014.78064.
4. A. Khatoon, Y. Hafeez, S Asghar and T. Ali "An Ontological Framework For Requirement Change Management in Distributed Environment" Published in The Nucleus A Quarterly Scientific Journal of Pakistan Atomic Energy Commission NCLEAM, ISSN 0 0 2 9 - 5 6 9 8" Volume 51, No. 2 (2014) PP: 291- 301 April 30, 2014.
5. M. Mukhtar, Y.Hafeez, M. Riaz, M.A. Khan,M. Ahmed, M.A. Abbas, H.Naz & A. Batool,"A Hybrid Model for Agile Practices Using Case Based Reasoning" IEEE 4thInternational Conference on Software Engineering and Service Science (ICSESS2013), May 23-25, 2013, Beijing, China. ISBN: 978-1-4673-5000-6/13/2013, PP: 820-823.
6. Y Hafeez, A batool,S Asghar,M Jamal "Role of Software Requirements to improve the Quality of Scrum Framework" Science International-Lahore ISI-indexed Journal, 26(1), 2014, ISSN# 1013-5316, CODEN: SINTE 8.
7. A. Batool, Y.Hafeez, S. Asghar, M.A. Abbas and M.S. Hassan, (2013). "A Scrum Framework for Requirement Engineering Practices "Published in Proceedings of the Pakistan Academy of Sciences, Volume 50, and Issue December, 2013: PP: 263-270 (2013) ISSN: 0377-2969.
8. M. Ahsan,Y.H. Motla and M., Asim " Knowledge Modelling for e-Agriculture using Ontology" 2014, 10th 2014 International Conference on Open Source Systems and

Technologies (ICOSST), December 19th and 20th 2014, Islamabad, Pakistan, ISBN: 978-1-4799-2054-9/14/2014.

9. S. Anwar,Y.H. Motla,Y. Sadiq, S., Asghar, M.,S.,Hassan and Z., I., Khan "User-Centred Design Practices in Scrum Development Process: A Distinctive Advantage?" 2014 IEEE 17th International Multi Topic Conference (INMIC 2014), December 8th-10th 2014, Bahria University, Karachi, Pakistan, ISBN: 978-1-4799-5754-5/14.

10. M., Khudad,Y., H., Motla, S., Asghar, S., A., Anwer and Z. , I., Khan "A Scrum Based Framework for E-Agriculture System" 2014 IEEE 17th International Multi Topic Conference (INMIC 2014), December 8th-10th 2014, Bahria University, Karachi, Pakistan, ISBN: 978-1-4799-5754-5/14.

11. M. Ahsan,Y.H. Motla, A., Anwer and M. ,W., Azeem "Knowledge Management Model for Support of Requirement Engineering" 2014 10th International conference on Emerging Technologies (ICET 2014), December 8th and 9th 2014, Islamabad, Pakistan,ISBN: 978-1-4799-6089-7/14.

12. S. Nazir,Y.H. Motla,T.,Abbas, A.,Khatoon,J., Jabeen, M.,Iqra & K. Bakhat"A Process Improvement in Requirement Verification and Validation using Ontology" 2014 IEEE Asia-Pacific World Congress on Computer Science and Engineering (APWC on CSE), November 4th and 5th 2014, Nadi,Fiji, DOI:10.1109/APWCCSE.2014.7053837,ISBN:978-1-4799-1955-0/14/2014.PP:1-8.

13. J., Jabeen, Y., H.,Motla, M., A., Abbasi, D., Batool, S., Nazir & S., A., Anwer " Incorporating Artificial Intelligence Technique into DSDM" 2014 IEEE Asia-Pacific World Congress on Computer Science and Engineering (APWC on CSE), November 4th and 5th 2014, Nadi,Fiji, DOI:10.1109/APWCCSE.2014.7053838,ISBN:978-1-4799-1955-0/14/2014.PP:1-8.

14. A. Akhater, Y. Hafeez Motla,H. Aslam and M. Jamal "Role of requirement change in software architecture using Twin Peaks Model" IEEE 5th International Conference on Software Engineering and Service Science (ICSESS2013) June 26-29, 2014, Beijing, China. ISBN: 978-1-4799-3278-8 /14/2014 IEEE, PP: 174 - 177.

15. A. Khatoon,Y.H. Motla,M., Azeem,H., Naz and S., Nazir "Requirement Change management for Global Software Development using ontology" 2013 9th International conference on Emerging Technologies (ICET 2013), December 9th and 10th 2013, Islamabad, Pakistan,ISBN: 978-1-4-4799-3456-0/13/2013.

16. H. Naz, Y. Hafeez, S.Asghar, M.Ahmed, M.S.Hassan, M. Mukhtar, A.Javed,"A Systematic Approach for Web Engineering Practices by Integrating Data Mining Technique with Requirement Change Management"IEEE 4th International

Conference on Software Engineering and Service Science (ICSESS2013) May 23-25, 2013, Beijing, China. ISBN: 978-1-4673-5000-6/13/2013 IEEE, PP: 373-376.

17. A. Batool, Y. Hafeez, B. Hamid, S. Asghar, M. Riaz, M. Mukhtar, and M. Ahmed. "Comparative Study of Traditional Requirement Engineering and Agile Requirement Engineering" The 15th International Conference on Advanced Communications Technology (ICACT 2013) Jan 27 – 30, 2013, at Phoenix Park, Korea, PP: 1006~1014, ISBN: 978-89-968650-0-1.

18. H. Naz, Y. Hafeez, S. Asghar, M.A. Abbas and A. Khatoon "Effective Usage of AI Technique for Requirement Change Management Practices "The 5th International Conference on Computer Science and Information Technology (CSIT2013) March 27-28, 2013, Amman, Jordan. ISBN -1-4673-5824-8.PP: 121-125.

19. Y. Hafeez, M. Riaz, S. Asghar, H. Naz, S.M.M. Gilani,A. Batool,M. Ahmed & M.S. Hassan. "A Requirement Change Management Framework for Distributed Software Environment"7th International Conference on Computing and Convergence Technology, (ICCCT), 2012 on 3-5 Dec., 2012, at Seoul, Korea (South). PP: 944-948, ISBN: 978-1-4673-0894-6.

20. Y. Hafeez, M.A. Abbas and G. Mustafa "Techniques for Data-Race Detection and Fault Tolerance: A Survey" International Conference on Computer & Information Sciences (ICCIS2012), 12-14 June 2012 at Kuala Lumpur Convention Centre Malaysia. ISBN: 978-1-4673-1938-6/12.

21. G. Mustafa, Y. Hafeez and M.A. Abbas "Fundamental Characteristics Creating Software Process Diversity" International Conference on Computer Network and Information Technology (ICCNIT11), 11-13th July, 2011, PP: 341-344. ISBN: 2223-6317.

References

[1] Šmite, Darja, Fabio Calefato, and Claes Wohlin. "Cost-Savings in Global Software Engineering: Where's the Evidence." *IEEE Software* 32, no. 4 (2015): 26-32.

[2] Hanssen, Geir K., Darja Smite, and Nils Brede Moe. "Signs of agile trends in global software engineering research: A tertiary study." In *Global Software Engineering Workshop (ICGSEW), 2011 Sixth IEEE International Conference on*, pp. 17-23. IEEE, 2011.

[3] Sutanto, Juliana, Atreyi Kankanhalli, and Bernard Cheng Yian Tan. "Investigating task coordination in globally dispersed teams: a structural contingency perspective." *ACM Transactions on Management Information Systems (TMIS)* 6, no. 2 (2015): 5.

[4] Richardson, Ita, Valentine Casey, Fergal McCaffery, John Burton, and Sarah Beecham. "A process framework for global software engineering teams."*Information and Software Technology* 54, no. 11 (2012): 1175-1191.

[5] Jain, Ritu, and Ugrasen Suman. "A Systematic Literature Review on Global Software Development Life Cycle." *ACM SIGSOFT Software Engineering Notes* 40, no. 2 (2015): 1-14.

[6] Paasivaara, Maria, and Casper Lassenius. "Communities of practice in a large distributed agile software development organization–Case Ericsson."*Information and Software Technology* 56, no. 12 (2014): 1556-1577.

[7] Mishra, Alok, and Deepti Mishra. "Cultural Issues in Distributed Software Development: A Review." In *On the Move to Meaningful Internet Systems: OTM 2014 Workshops*, pp. 448-456. Springer Berlin Heidelberg, 2014.

[8] Colomo-Palacios, Ricardo, Cristina Casado-Lumbreras, Pedro Soto-Acosta, Francisco José García-Peñalvo, and Edmundo Tovar. "Project managers in global software development teams: a study of the effects on productivity and performance." *Software Quality Journal* 22, no. 1 (2014): 3-19.

[9] Verner, June M., O. Pearl Brereton, Barbara A. Kitchenham, Mahmood Turner, and Mahmood Niazi. "Risks and risk mitigation in global software development: A tertiary study." *Information and Software Technology* 56, no. 1 (2014): 54-78.

[10] Pacheco, C. L., I. A. Garcia, José Antonio Calvo-Manzano, and Magdalena Arcilla. "A proposed model for reuse of software requirements in requirements catalog." *Journal of Software: Evolution and Process* 27, no. 1 (2015): 1-21.

[11] Standish. CHAOS Manifesto: The Laws of CHAOS and the CHAOS 100 Best PM Practices, *The Standish Group International Inc., Boston*, 2013.

[12] Aitken, Ashley, and Vishnu Ilango. "A comparative analysis of traditional software engineering and agile software development." In *System Sciences (HICSS), 2013 46th Hawaii International Conference on*, pp. 4751-4760. IEEE, 2013.

[13] Dingsøyr, Torgeir, Sridhar Nerur, VenuGopal Balijepally, and Nils Brede Moe. "A decade of agile methodologies: Towards explaining agile software development." *Journal of Systems and Software* 85, no. 6 (2012): 1213-1221.

[14] Hossain, Emam, Muhammed Ali Babar, Hye-young Paik, and June Verner. "Risk identification and mitigation processes for using scrum in global software development: A conceptual framework." In *Software Engineering Conference, 2009. APSEC'09. Asia-Pacific*, pp. 457-464. IEEE, 2009.

[15] Cocco, Luisanna, Katiuscia Mannaro, and Giulio Concas. "A Model for Global Software Development with Cloud Platforms." In *Software Engineering and Advanced Applications (SEAA), 2012 38th EUROMICRO Conference on*, pp. 446-452. IEEE, 2012.

[16] Jalali, Samireh, and Claes Wohlin. "Agile practices in global software engineering-A systematic map." In *Global Software Engineering (ICGSE), 2010 5th IEEE International Conference on*, pp. 45-54. IEEE, 2010.

[17] Valentine Casey. 2011. Imparting the importance of culture to global software development. *ACM Inroads* 1, 3 (September 2011), pp. 51-57.

[18] Šmite, Darja, Claes Wohlin, Tony Gorschek, and Robert Feldt. "Empirical evidence in global software engineering: a systematic review." *Empirical software engineering* 15, no. 1 (2010): 91-118.

[19] Berger, Christian, and Ulrik Eklund. "Expectations and Challenges from Scaling Agile in Mechatronics-Driven Companies–A Comparative Case Study." In *Agile Processes, in Software Engineering, and Extreme Programming*, pp. 15-26. Springer International Publishing, 2015.

[20] Harmsen, Anton Frank, J. N. Brinkkemper, and JL Han Oei. *Situational method engineering for information system project approaches*. University of Twente, Department of Computer Science, 1994.

[21] Abad, Zahra Shakeri Hossein, Anahita Alipour, and Raman Ramsin. "Enhancing Tool Support for Situational Engineering of Agile Methodologies in Eclipse." In *Software Engineering Research, Management and Applications 2012*, pp. 141-152. Springer Berlin Heidelberg, 2012.

[22] Yaser Hafeez, A batool, Sohail Asghar,M Jamal "Role of Software Requirements to improve the Quality of Scrum Framework" Science International Lahore ISI-indexed Journal, 26(1), 2014, ISSN# 1013-5316, CODEN: SINTE 8.

[23] Omoronyia, Inah, John Ferguson, Marc Roper, and Murray Wood. "A review of awareness in distributed collaborative software engineering." *Software: Practice and Experience* 40, no. 12 (2010): 1107-1133.

[24] Aranda, Gabriela N., Aurora Vizcaíno, and Mario Piattini. "Analyzing and Evaluating the Main Factors that Challenge Global Software Development."*Open Software Engineering Journal* 4, no. 1 (2010): 14-25.

[25] Bannerman, Paul L., Emam Hossain, and Ross Jeffery. "Scrum practice mitigation of global software development coordination challenges: a distinctive advantage?" In *System Science (HICSS), 2012 45th Hawaii International Conference on*, pp. 5309-5318. IEEE, 2012.

[26] S. Sultana, Y.Hafeez , S. Asghar, M. Jamal and R. Azad "A Hybrid Model by Integrating Agile Practices for Pakistani Software Industry "Accepted in 24th International Conference on Electronics, Communications and Computers", CONIELECOMP ISBN: 978-1-4799-3469-0/14, 2014 Cholula, Puebla, MEXICO.

[27] Meijer, P.C., N. Verloop. and D. Beijaard. 2002. Multi-method triangulation in a qualitative study on teachers' practical knowledge: An attempt to increase internal validity. Quality and Quantity, 36(2), pp.145–167.

[28] Johnson, R.B., a.J. Onwuegbuzie. and L.a. Turner. 2007. Toward a Definition of Mixed Methods Research. Journal of Mixed Methods Research, 1(2), pp.112–133.

[29] Miller, James. "Triangulation as a basis for knowledge discovery in software engineering." Empirical Software Engineering 13, no. 2 (2008): 223-228.

[30] Adolph, Steve, Wendy Hall, and Philippe Kruchten. "Using grounded theory to study the experience of software development." Empirical Software Engineering 16, no. 4 (2011): 487-513.

[31] Miller, J. 2008. Triangulation as a basis for knowledge discovery in software engineering. Empirical Software Engineering, 13(2), pp.223–228.

[32] Wohlin, Claes, Martin Höst, and Kennet Henningsson. "Empirical research methods in software engineering." In Empirical methods and studies in software engineering, pp. 7-23. Springer Berlin Heidelberg, 2003.

[33] Zelkowitz, Marvin V. "Techniques for Empirical validation." In Empirical Software Engineering Issues. Critical Assessment and Future Directions, pp. 4-9. Springer Berlin Heidelberg, 2007.

[34] Dyba, T., Kampenes, V. B. & Sjøberg, D. I. K. (2006), 'A systematic review of Statistical power in software engineering experiments', Information & Software Technology 48(8), 745–755.

[35] Glass, R. L., Vessey, I. & Ramesh, V. (2002), 'Research in software engineering: an analysis of the literature', Information & Software Technology 44(8), 491–506.

[36] Kitchenham, B. & Charters, S. (2007), Guidelines for performing systematic literature reviews in software engineering, Technical Report EBSE-2007-01, School of Computer Science and Mathematics, Keele University.

[37] Dybå, Tore, and Torgeir Dingsøyr. "Empirical studies of agile software development: A systematic review." Information and software technology 50, no. 9 (2008): 833-859.

[38] Portillo-Rodríguez, J., A. Vizcaíno., C. Ebert. And M. Piattini. 2010. Tools to support global software development processes: a survey. In Global Software Engineering (ICGSE), 2010 5th IEEE International Conference on pp. 13-22.

[39] da Silva, Fabio QB, Catarina Costa, A. Cesar C. Franca, and Rafael Prikladinicki. "Challenges and solutions in distributed software development project management: a systematic literature review." In Global Software Engineering (ICGSE), 2010 5th IEEE International Conference on, pp. 87-96. IEEE, 2010.

[40] Niazi, Mahmood, Naveed Ikram, Muneera Bano, Syed Imtiaz, and Samee U. Khan. "Establishing trust in offshore software outsourcing relationships: an exploratory study using a systematic literature review." Software, IET 7, no. 5 (2013): 283-293.

[41] Jalali, Samireh, and Claes Wohlin. "Global software engineering and agile practices: a systematic review." Journal of Software: Evolution and Process24, no. 6 (2012): 643-659.

[42] Tariq, A. and A.A. Khan. 2012. Framework supporting team and project activities in Global Software Development (GSD). Proceedings - 2012 International Conference on Emerging Technologies, ICET 2012, pp.334–339.

[43] Beecham, Sarah, Tony Clear, John Barr, and John Noll. "Challenges and Recommendations for the Design and Conduct of Global Software Engineering Courses: A Systematic Review Protocol."

[44] Shrivastava, Suprika Vasudeva, and Urvashi Rathod. "Risks in distributed agile development: A review." Procedia-Social and Behavioral Sciences 133 (2014): 417-424.

[45] Christopher, Martin, Carlos Mena, Omera Khan, and Oznur Yurt. "Approaches to managing global sourcing risk." Supply Chain Management: An International Journal 16, no. 2 (2011): 67-81.

[46] Šmite, Darja, Claes Wohlin, Zanc Galviņa, and Rafael Prikladnicki. "An empirically based terminology and taxonomy for global software engineering." Empirical Software Engineering 19, no. 1 (2014): 105-153.

[47] Nuevo, Eva del, Mario Piattini, and Francisco J. Pino. "Scrum-based methodology for distributed software development." In Global Software Engineering (ICGSE), 2011 6th IEEE International Conference on, pp. 66-74. IEEE, 2011.

[48] Hossain, Emam, Muhammad Ali Babar, and Hye-young Paik. "Using scrum in global software development: a systematic literature review." In Global Software Engineering, 2009. ICGSE 2009. Fourth IEEE International Conference on, pp. 175-184. Ieee, 2009.

[49] S. Anwar,Y.H. Motla,Y. Sadiq, S., Asghar, M.,S.,Hassan and Z., I., Khan "User-Centred Design Practices in Scrum Development Process: A Distinctive Advantage?" 2014 IEEE 17th International Multi Topic Conference (INMIC 2014), December 8th-10th 2014, Bahria University, Karachi, Pakistan, ISBN: 978-1-4799-5754-5/14.

[50] Qahtani, A., W. Gary. and G. Andrew. 2012. Toward a framework for development and specialisation of product software across organisational boundaries. , pp.391–395.

[51] Kaskenpalo, Petteri, and Stephen G. MacDonell. "Valuing evaluation: methodologies to bridge research and practice." In Proceedings of the 2nd international workshop on evidential assessment of software technologies, pp. 27-32. ACM, 2012.

[52] D.S. Cruzes, T. Dybå. Synthesizing Evidence in Software Engineering Research. In Proceedings of the 4th International Symposium on Empirical Software Engineering and Measurement (ESEM 2010), ACM, 2010.

[53] Richardson, Ita, Valentine Casey, Fergal McCaffery, John Burton, and Sarah Beecham. "A process framework for global software engineering teams."Information and Software Technology 54, no. 11 (2012): 1175-1191.

[54] Qureshi, M. Rizwan Jameel, and Isaac Sayid. "Scheme of Global Scrum Management Software." (2015) I.J. Information Engineering and Electronic Business, 2015, 2, pp. 1-7.

[55] A. Sarwar,k.,Bhakat and Y. Hafeez "An A Situational Requirement Engineering Model for an Agile Process Ontological Framework For Requirement Change Management In Distributed Environment" Journal of information and Communication Technologies ISSN 0 0 2 9 - 5 6 9 8" Volume 51, No. 2 (2015).

[56] Yildiz, Bugra M., and Bedir Tekinerdogan. "Architectural viewpoints for global software development." In Global Software Engineering Workshop (ICGSEW), 2011 Sixth IEEE International Conference on, pp. 9-16. IEEE, 2011.

[57] Pocatilu, P., F. Alecu. and M. Vetrici. 2010. Measuring the Efficiency of Cloud Computing for E-learning Systems 2 Cloud Computing., 9(1), pp.42–51.

[58] Paasivaara, Maria, and Casper Lassenius. "Could global software development benefit from agile methods?." In Global Software Engineering, 2006. ICGSE'06. International Conference on, pp. 109-113. IEEE, 2006.

[59] Nurdiani, Indira, Ronald Jabangwe, Darja Smite, and Daniela Damian. "Risk identification and risk mitigation instruments for global software development: Systematic review and survey results." In Global Software Engineering Workshop (ICGSEW), 2011 Sixth IEEE International Conference on, pp. 36-41. IEEE, 2011.

[60] Dorairaj, Siva, James Noble, and Petra Malik. "Effective communication in distributed Agile software development teams." In Agile Processes in Software Engineering and Extreme Programming, pp. 102-116. Springer Berlin Heidelberg, 2011.

[61] A., Sarwar, Y., Hafeez, S., Asghar, K., war and Z., Gul (2015). "Outsourcing Agile Software Development Model Using Grounded Theory" Published in Journal of Applied Environmental and Biological Sciences (JAEBS) (Thomson Reuters ISI indexed), 2015, J.Appl. Environ. Biol. Sci., Volume: 5(12)1-1, 2015, ISSN: 2090-4274.

[62] Šmite, Darja, and Claes Wohlin. "Software product transfers: Lessons learned from a case study." In Global Software Engineering (ICGSE), 2010 5th IEEE International Conference on, pp. 97-105. IEEE, 2010.

[63] Mnkandla, Ernest, and Barry Dwolatzky. "Agile methodologies selection toolbox." In Software Engineering Advances, 2007. ICSEA 2007. International Conference on, pp. 72-72. IEEE, 2007.

[64] Mikulėnas, Gytenis, Rimantas Butleris, and Lina Nemuraitė. "An approach for the metamodel of the framework for a partial agile method adaptation."Information Technology And Cóntrol 40, no. 1 (2011): 71-82.

[65] Ayed, Hedi, Benoiit Vanderose, and Naji Habra. "A metamodel-based approach for customizing and assessing agile methods." In Quality of Information and Communications Technology (QUATIC), 2012 Eighth International Conference on the, pp. 66-74. IEEE, 2012.

[66] JA Zachman, The Zachman Framework For Enterprise Architecture: Primer for Enterprise Engineering and Manufacturing , 2003, Zachman International OMG BRWG's RFI.

[67] Van-De-Weerd, I., S. Brinkkemper, J. Souer and J. Versendaal. 2006. A Situational Implementation Method for Web-based Content Management System-applications: Method Engineering and Validation in Practice. Software Process: Improvement and Practice (11:5), pp. 521-538.

[68] ISO/IEC.2014. Software Engineering-Metamodel for Development Methodologies. ISO/IEC 24744. International Standards Organization/International Electrotechnical Commission, Geneva

[69] Matthiesen, Stina. "Global Software Development: Exploring Multiplicity and Asymmetric Dynamics in Collaborative Work." In Proceedings of the 19th ACM

Conference on Computer Supported Cooperative Work and Social Computing Companion, pp. 163-166. ACM, 2016.

[70] J., Jabeen, Y.,Hafeez, M., and A., Anwer "Incorporating Artificial Intelligence Technique into DSDM" 2014 IEEE Asia-Pacific World Congress on Computer Science and Engineering (APWC on CSE), 2014, Nadi,Fiji, ISBN:978-1-4799-1950/14/2014.

[71] Xia, Hao, Milind Dawande, and Vijay Mookerjee. "Optimal Coordination in Distributed Software Development." Production and Operations Management25, no. 1 (2016): 56-76.

[72] Bhatt, Punita, Ali J. Ahmad, and Muhammad Azam Roomi. "Social innovation with open source software: User engagement and development challenges in India." Technovation (2016).

[73] Umar, Muhammad Aminu, Sheidu Salami Tenuche, Sahabi Ali Yusuf, Aminu Onimisi Abdulsalami, and Aliyu Muhammad Kufena. "Usability Engineering in Agile Software Development Processes." Emerging Innovations in Agile Software Development (2016): 208.

[74] Ambler, Scott W., and Mark Lines. "The Disciplined Agile Process Decision Framework." In Software Quality. The Future of Systems-and Software Development, pp. 3-14. Springer International Publishing, 2016.

[75] M. Mukhtar, Y.Hafeez, M. Riaz, M.A. Khan,M. Ahmed, M.A. Abbas, H.Naz & A. Batool,"A Hybrid Model for Agile Practices Using Case Based Reasoning" IEEE 4thInternational Conference on Software Engineering and Service Science (ICSESS2013), May 23-25, 2013, Beijing, China. ISBN: 978-1-4673-5000-6/13/2013, PP: 820-823.

[76] Benbasat, Izak, David K. Goldstein, and Melissa Mead. "The case research strategy in studies of information systems." MIS quarterly (1987): 369-386.

[77] Gumm, Dorina C. "Distribution dimensions in software development projects: a taxonomy." Software, IEEE 23, no. 5 (2006): 45-51.

[78] Chuang, Shui-Lung, and Lee-Feng Chien. "Automatic query taxonomy generation for information retrieval applications." Online Information Review27, no. 4 (2003): 243-255.

[79] Fitzgerald, Brian, and Par J. Agerfalk. "Outsourcing to an unknown workforce: exploring opensourcing as a global sourcing strategy." MIS Quarterly 32, no. 2 (2010): 385-409.

[80] Höfner, Peter, Ridha Khedri, and Bernhard Möller. "An algebra of product families." Software & Systems Modeling 10, no. 2 (2011): 161-182.

[81] Prikladnicki, Rafael, and Jorge Luis Nicolas Audy. "Process models in the practice of distributed software development: A systematic review of the literature." Information and Software Technology 52, no. 8 (2010): 779-791.

[82] Bock, Florian, Daniel Homm, Sebastian Siegl, and Reinhard German. "A Taxonomy for Tools, Processes and Languages in Automotive Software Engineering." arXiv preprint arXiv:1601.03528 (2016).

[83] Alonso Gaona García, Paulo, David Martín-Moncunill, Salvador Sánchez-Alonso, and Ana Fermoso García. "A usability study of taxonomy visualisation user interfaces in digital repositories." Online Information Review 38, no. 2 (2014): 284-304.

[84] Dingsøyr, Torgeir, Tor Erlend Fægri, and Juha Itkonen. "What is large in large-scale? A taxonomy of scale for agile software development." InProduct-Focused Software Process Improvement, pp. 273-276. Springer International Publishing, 2014.

[85] Carmel, Erran, and Pamela Abbott. "Why 'nearshore' means that distance matters." Communications of the ACM 50, no. 10 (2007): 40-46.

[86] Malone, Thomas W. What is coordination theory? Cambridge, Mass.: Massachusetts Institute of Technology, 1988.

[87] Darja Šmite & Claes Wohlin & Zane Galviņa & Rafael Prikladnicki "An empirically based terminology and taxonomy for global software engineering." Springer Science, LLC 2012.

[88] Ayed, et al. " AM-QuICk : a measurement-based framework for agile methods Customization." In International Conference on Software Process and Product Measurement 2013, pp: 71-80. IEEE Computer Society, 2013.

[89] Gregor, Shirley. "The nature of theory in information systems." MIS quarterly(2006): 611-642.

[90] Van de Ven, Andrew H., Andre L. Delbecq, and Richard Koenig Jr. "Determinants of coordination modes within organizations." American sociological review (1976): 322-338.

[91] Okhuysen, Gerardo A., and Beth A. Bechky. "10 coordination in organizations: an integrative perspective." The Academy of Management Annals 3, no. 1 (2009): 463-502.

[92] Arrow, Holly, Joseph E. McGrath, and Jennifer L. Berdahl. Small groups as complex systems: Formation, coordination, development, and adaptation. Sage Publications, 2000.

[93] Kotlarsky, Julia, Paul C. Van Fenema, and Leslie P. Willcocks. "Developing a knowledge-based perspective on coordination: The case of global software projects." Information & Management 45, no. 2 (2008): 96-108.

[94] Gumm, Dorina C. "Distribution dimensions in software development projects: a taxonomy." Software, IEEE 23, no. 5 (2006): 45-51.

[95] Heath, Helen, and Sarah Cowley. "Developing a grounded theory approach: a comparison of Glaser and Strauss." International journal of nursing studies41, no. 2 (2004): 141-150.

[96] Dan, Ph D. "Success Factors That Influence Agile Software Development Project Success." American Scientific Research Journal for Engineering, Technology, and Sciences (ASRJETS) 17, no. 1 (2016): 172-222.

[97] Nguyen, Tuan Manh. "A Systems Theory of Organizational Information."International Journal of Knowledge and Systems Science (IJKSS) 7, no. 2 (2016): 58-83.

[98] Gandomani, Taghi Javdani, and Mina Ziaei Nafchi. "An empirically-developed framework for Agile transition and adoption: A Grounded Theory approach."Journal of Systems and Software 107 (2015): 204-219.

[99] Robert K.. Yin. Case study research: Design and methods. Sage, 2003.

[100] Dingsøyr, Torgeir, Sridhar Nerur, VenuGopal Balijepally, and Nils Brede Moe. "A decade of agile methodologies: Towards explaining agile software development." Journal of Systems and Software 85, no. 6 (2012): 1213-1221.

[101] Hossain, Emam, Muhammad Ali Babar, and June Verner. "Towards a framework for using agile approaches in global software development." InProduct-Focused Software Process Improvement, pp. 126-140. Springer Berlin Heidelberg, 2009.

[102] Zelkowitz, M. V. 2007. Techniques for Empirical validation. In Empirical Software Engineering Issues. Critical Assessment and Future Directions (pp. 4-9).

[103] Jain, Ritu, and Ugrasen Suman. "A Systematic Literature Review on Global Software Development Life Cycle." ACM SIGSOFT Software Engineering Notes 40, no. 2 (2015): 1-14.

[104] Prikladnicki, R., A. Boden., G. Avram., C.R. de Souza. and V. Wulf. 2014. Data collection in global software engineering research: learning from past experience. Empirical Software Engineering, 19(4), 822-856.

[105] Stavru, Stavros, and Sylvia Ilieva. "Agile and the Global Software Leaders: A Perfect Match?." In Agile Processes, in Software Engineering, and Extreme Programming, pp. 230-235. Springer International Publishing, 2015.

[106] Yang, Qing, Sonia Kherbachi, Yoo Suk Hong, and Chen Shan. "Identifying and managing coordination complexity in global product development project." International Journal of Project Management 33, no. 7 (2015): 1464-1475.

[107] Davison, Robert, Maris G. Martinsons, and Ned Kock. "Principles of canonical action research." Information systems journal 14, no. 1 (2004): 65-86.

[108] Jung, Ho-Won, Seung-Gweon Kim, and Chang-Shin Chung. "Measuring software product quality: A survey of ISO/IEC 9126." IEEE software 5 (2004): 88-92.

[109] Zachman, John A. "Zachman framework." John A. Zachman and Zachman International, Inc 201, no. 2 (2012).

[110] Zachman, Angela L., Spencer W. Crowder, Ophir Ortiz, Katarzyna J. Zienkiewicz, Christine M. Bronikowski, Shann S. Yu, Todd D. Giorgio, Scott A. Guelcher, Joachim Kohn, and Hak-Joon Sung. "Pro-angiogenic and anti-inflammatory regulation by functional peptides loaded in polymeric implants for soft tissue regeneration." Tissue engineering Part A 19, no. 3-4 (2012): 437-447.

[111] Henderson-Sellers, B. and M.K. Serour. 2005. Creating a Dual Agility Method – The Value of Method Engineering. Journal of Database Management 16(4), pp.1-24.

[112] Henderson-Sellers, B. 2006. Method Engineering: Theory and Practice. In Information Systems Technology and Its Applications. 5th International Conference ISTA 2006. pp. 13-23.

[113] Henderson-Sellers, Brian, and Jolita Ralyté. "Situational Method Engineering: State-of-the-Art Review." J. UCS 16, no. 3 (2010): 424-478.

[114] Kornyshova, Elena, Rébecca Deneckère, and Camille Salinesi. "Method chunks selection by multicriteria techniques: an extension of the assembly-based approach." In Situational Method Engineering: Fundamentals and Experiences, pp. 64-78. Springer US, 2007.

[115] Lal, Ramesh. "Strategic factors in agile software development method adaptation: a study of market-driven organisations: a thesis presented in partial [fulfilment] of the requirements for the degree of Doctor of Philosophy in Information Technology at Massey University, Albany campus, New Zealand." PhD diss., 2011.

[116] Flatscher, Rony G. "Metamodeling in EIA/CDIF---meta-metamodel and metamodels." ACM Transactions on Modeling and Computer Simulation (TOMACS) 12, no. 4 (2002): 322-342.

[117] Fitzgerald, Brian, Gerard Hartnett, and Kieran Conboy. "Customising agile methods to software practices at Intel Shannon." European Journal of Information Systems 15, no. 2 (2006): 200-213.

[118] Brinkkemper, Sjaak. "Method engineering: engineering of information systems development methods and tools." Information and software technology 38, no. 4 (1996): 275-280.

[119] Rolland, Colette, and Naveen Prakash. "A proposal for context-specific method engineering." In Method Engineering, pp. 191-208. Springer US, 1996.

[120] Mirbel, Isabelle, and Jolita Ralyté. "Situational method engineering: combining assembly-based and roadmap-driven approaches." Requirements Engineering 11, no. 1 (2006): 58-78.

[121] Ågerfalk, P., and Brian Fitzgerald. "Exploring the concept of method rationale: A conceptual tool." Advanced topics in database research 5 (2006): 63-78.

[122] Runeson, P. and M. Höst. 2009. Guidelines for conducting and reporting case study research in software engineering. Empirical Software Engineering, 14, pp.131–164.

[123] Saeki, Motoshi. "CAME: The first step to automated method engineering." In Workshop on Process Engineering for Object-Oriented and Component-Based Development, Anaheim, CA. 2003.

[124] Dahanayake, A. N. W. "Evaluation of the strength of computer aided method engineering for product development process modeling." In Database and Expert Systems Applications, pp. 394-410. Springer Berlin Heidelberg, 1998.

[125] Mirakhorli, Mehdi, Abdorrahman Khanipour Rad, Fereidoon Shams Aliee, Abbas Mirakhorli, and Maryam Pazoki. "Rdp technique: Take a different look at xp for adoption." In Software Engineering, 2008. ASWEC 2008. 19th Australian Conference on, pp. 656-662. IEEE, 2008.

[126] Runeson, Per, and Martin Höst. "Guidelines for conducting and reporting case study research in software engineering." Empirical software engineering14, no. 2 (2009): 131-164.

[127] Damian, Daniela E., and Didar Zowghi. "RE challenges in multi-site software development organizations." Requirements engineering 8, no. 3 (2003): 149-160.

[128] Lagerberg, Lina, and Tor Skude. "The impact of agile principles and practices on large-scale software development projects: A multiple-case study of two software development projects at Ericsson." (2013).

[129] Dorairaj, Siva, James Noble, and Petra Malik. "Knowledge management in distributed agile software development." In Agile Conference (AGILE), 2012, pp. 64-73. IEEE, 2012.

[130] Bass, Julian M. "Influences on agile practice tailoring in enterprise software development." In AGILE India (AGILE INDIA), 2012, pp. 1-9. IEEE, 2012.

[131] Khan, Siffat Ullah. "Software outsourcing vendors' readiness model (SOVRM)." PhD diss., Keele University, 2011.

[132] Paasivaara, Maria, Casper Lassenius, Ville T. Heikkila, Kim Dikert, and Christian Engblom. "Integrating global sites into the lean and agile transformation at ericsson." In Global Software Engineering (ICGSE), 2013 IEEE 8th International Conference on, pp. 134-143. IEEE, 2013.

[133] Phalnikar, Rashmi, V. S. Deshpande, and S. D. Joshi. "Applying agile principles for distributed software development." In Advanced Computer Control, 2009. ICACC'09. International Conference on, pp. 535-539. IEEE, 2009.

[134] Kitchenham, Barbara, Lesley Pickard, and Shari Lawrence Pfleeger. "Case studies for method and tool evaluation." IEEE software 12, no. 4 (1995): 52.

[135] Ambler, Scott W., and Mark Lines. "The Disciplined Agile Process Decision Framework." In Software Quality. The Future of Systems-and Software Development, pp. 3-14. Springer International Publishing, 2016.

[136] Levy, Y., & Ellis, T. J. (2011). A guide for novice researchers on experimental and Quasi-experimental studies in information systems research. Interdisciplinary Journal of Information, Knowledge, and Management, 6, 151–161.

[137] Popescu, E. (2010). Adaptation provisioning with respect to learning styles in a Web-based educational system: an experimental study. Journal of Computer Assisted Learning, 26(4), 243–257. doi:10.1111/j.1365-2729.2010.00364.

[138] Fitrisia, Y., & Hendradjaya, B. (2014). Implementation of ISO 9126-1 quality model for asset inventory information system by utilizing object oriented metrics. In Electrical Engineering and Computer Science (ICEECS), 2014 International Conference on (pp. 229–234).

[139] Ko, A.J., T.D. LaToza. and M.M. Burnett. 2013. A practical guide to controlled experiments of software engineering tools with human participants. Empirical Software Engineering, pp.1–32.

Appendix A

Situational Agile Framework Evaluation (Participants Feedback)

DISCLAMIMER: Information gathered from this questionnaire will strictly be confidential. Entire information will be used for research purpose only and will not be shared with third party under any circumstances. Responses shall not be disclosed in any output form from this research.

Objective of the Experimental Study: The purpose of the current study is to investigate Agile situational framework, the knowledge level and understanding of the participants about the situations handling during Agile GSD development.

Form#________

DEMOGRAPHIC INFORMATION

This information is confidential and will be used for interpreting results.

1. Please provide approximate answers to the following questions.

a). Your present position – Title: _____ b) Number of years in the company: _____

c). Your experience in Agile software development related jobs _____Years

d). Highest level of education completed: _____

2. Is your company involved in developing Agile software development in GSD environment?

☐ Yes ☐ No

3. Has your organization been adopted for pre development, development and post development Activities for Agile GSD development?

☐ Yes ☐ No

4. How many (approximately) staff are employed by your organization?

☐ < = 10 ☐ 11 – 50 ☐ 51 – 100 ☐ 101- 200 ☐ 200 – 500 ☐ > 500

Terms (Glossary):

SME: Situational Method Engineering, GSD: Global Software Development.

Context: Agile Software Development Process Improvement in GSD.

Q1. Why did your company embark on Agile GSD process improvement programme? You may tick only one option and rate the reasons on scale from 5-1 where 5 is the most important and 1 is the least important reason for starting from Agile GSD development.

What are the main reason for starting Situational Agile GSD development in your organization?	1	2	3	4	5
To Reduce the cost in software development	☐	☐	☐	☐	☐
Improvement of cycle times for software development	☐	☐	☐	☐	☐
To shorten the time-to-market in GSD Environment	☐	☐	☐	☐	☐
To enhance productivity in Agile GSD software development	☐	☐	☐	☐	☐
To improve quality of the software developed	☐	☐	☐	☐	☐
To fulfill customers' needs as per specification	☐	☐	☐	☐	☐
To avoid the rework time and to make forecasting better for schedule	☐	☐	☐	☐	☐

Q2. OVERALL, HOW WOULD YOU ELABORATE SITUATIONAL AGILE METHOD PLAY VITAL ROLE IN SOFTWARE PROCESS IMPROVEMENT AT YOUR COMPANY?

Issue	**1**	**2**	**3**	**4**	**5**
Selection of appropriate fragment is a lengthy process than we expected	☐	☐	☐	☐	☐
Designing of a new Agile framework in GSD is more costly and training of manpower is required	☐	☐	☐	☐	☐
Technical support and skill is required to assemble method fragments	☐	☐	☐	☐	☐
Deploying a new framework is a difficult due to time and resources restrictions	☐	☐	☐	☐	☐

Q3. WHICH OF THE FOLLOWING SUCCESS FACTORS HAVE CONTRIBUTED TO THE SUCCESS OF 3C-SAME FRAMEWORK?

Success Factor	**Level of contribution**				
	1	**2**	**3**	**4**	**5**
Assignment of responsibility of Method Engineers and GSD Managers	☐	☐	☐	☐	☐
Clear and relevant SME principles	☐	☐	☐	☐	☐
Customer satisfaction is directly associated with experienced staff	☐	☐	☐	☐	☐
Defined Agile in GSD implementation methodology	☐	☐	☐	☐	☐
Encouraging communication and coordination among GSD Teams	☐	☐	☐	☐	☐
Light weight documentation	☐	☐	☐	☐	☐
Higher staff moral with encouragement and internal leadership	☐	☐	☐	☐	☐
Providing enhanced understanding between GSD Teams in order to provide logical order or theoretical steps into practices	☐	☐	☐	☐	☐
Standards and procedures for the construction of situational methodology	☐	☐	☐	☐	☐
Tailoring improvement initiatives to customize the Agile GSD	☐	☐	☐	☐	☐

Q4. WHICH OF THE FOLLOWING CHALLENGES MAY HAVE NEGATIVE EFFECT ON THE SUCCESS OF 3C-SAME FRAMEWORK?

Barrier	**Level of effect**				
	1	**2**	**3**	**4**	**5**
Inexperienced staff/lack of domain knowledge	☐	☐	☐	☐	☐
Lack of Agile awareness	☐	☐	☐	☐	☐
Lack of communication coordination and control	☐	☐	☐	☐	☐
Lack of defined SME implementation methodology	☐	☐	☐	☐	☐
Lack of tools and training	☐	☐	☐	☐	☐
Negative/Bad experience	☐	☐	☐	☐	☐

Appendix B

Situational Agile Framework Evaluation (Interview Questions)

DISCLAMIMER: Information gathered from this questionnaire will strictly be confidential. Entire information will be used for research purpose only and will not be shared with third party under any circumstances. Responses shall not be disclosed in any output form from this research.

Objective of the Experimental Study: The purpose of the current study is to conduct interview from the focused group to investigate Agile situational framework. Furthermore, we evaluate the framework using ISO 9126-7 quality model.

Q1. Do you consider 3C-SAME Framework a suitable approach to improving the quality of the software product?

☐ Exist ☐ Partially Exist ☐ Does not Exist

Q2. Has the 3C-SAME Framework provided constructed situational methodology expected and clear benefits to management?

☐ Exist ☐ Partially Exist ☐ Does not Exist

Q3. Does the 3C-SAME Framework is suitable in order to formulate the situational method to improve the productivity?

☐ Exist ☐ Partially Exist ☐ Does not Exist

Q4. How this 3C-SAME framework be applied in method construction to reduce work dependencies and improve the accuracy?

☐ Exist ☐ Partially Exist ☐ Does not Exist

Q5. Would the 3C-SAME framework provide mechanism for usability to the GSD managers in order to provide the ease of use?

☐ Exist ☐ Partially Exist ☐ Does not Exist

Q6. What kind of projects are more appropriate for customization which ultimately enhance overall quality in terms of time behavior?

☐ Exist ☐ Partially Exist ☐ Does not Exist

Q7. Does the 3C-SAME framework have the capability to method enactment for Agile GSD teams?

☐ Exist ☐ Partially Exist ☐ Does not Exist

Q8. Does the meta model compatible for 3C-SAME framework in order to formulate new method?

☐ Exist ☐ Partially Exist ☐ Does not Exist

Q9. Does tool provide a mechanism to support method engineer and GSD managers in order to facilitate interactive communication within GSD teams?

☐ Exist ☐ Partially Exist ☐ Does not Exist

Q10. Does joint retrospective facilitate the experience factory to reuse the existing fragment?

☐ Exist ☐ Partially Exist ☐ Does not Exist

Appendix C

RAW DATA OF PRE-TEST SCORES

Control Group (N=45)

		Familiar Content															Unseen Contents												Result			
Experience	Participants	Q 1.1	Q 1.2	Q 1.3	Q 1.4	Q 1.5	Q 1.6	Q 1.7	Q 2.1	Q 2.2	Q 2.3	Q 2.4	Q 3.1	Q 3.2	Q 3.3	Q 3.4	Q 3.5	Q 3.6	Q 3.7	Q 3.8	Q 3.9	Q 3.10	Q 4.1	Q 4.2	Q 4.3	Q 4.4	Q 4.5	Q 4.6	Total	Mean of Familiar	Mean of Unseen	Total Mean
±2 Years (N=16)	Participant 1	1	2	3	1	1	1	2	1	1	2	3	2	1	2	3	2	1	1	1	1	2	2	1	1	1	1	2	31	1.58	1.56	3.14
	Participant 2	2	1	1	4	1	1	2	4	3	2	2	1	1	1	2	1	1	1	2	1	1	2	2	1	1	4	2	33			
	Participant 3	1	3	1	1	2	1	1	1	1	2	1	2	2	2	1	4	1	3	1	1	1	1	2	3	1	1	3	31			
	Participant 4	1	2	2	2	3	1	1	1	2	2	1	1	1	1	1	1	1	3	2	2	1	3	1	1	1	2	4	29			
	Participant 5	3	1	1	2	2	3	1	1	1	2	2	1	1	4	1	1	1	2	1	2	1	1	3	1	2	2	2	31			
	Participant 6	4	1	4	2	1	2	1	1	1	1	2	3	1	1	2	2	2	1	1	2	1	2	3	3	1	1	1	33			
	Participant 7	1	2	3	2	1	1	2	2	1	3	1	1	1	2	1	1	1	2	2	2	1	1	2	2	1	2	1	30			
	Participant 8	2	1	3	1	1	1	1	2	1	1	3	1	2	2	1	1	1	2	1	1	2	1	2	1	1	2	2	28			
	Participant 9	3	1	1	1	2	1	1	2	1	2	3	3	1	1	1	1	2	1	1	1	2	2	1	1	1	1	3	29			
	Participant 10	1	1	4	1	1	1	2	2	1	1	2	2	2	2	1	1	1	2	2	1	1	2	2	1	1	4	3	30			
	Participant 11	1	2	1	1	1	2	1	1	2	1	2	1	1	2	2	2	1	1	1	1	3	1	2	3	1	1	2	26			
	Participant 12	1	2	1	3	1	3	2	1	1	1	2	2	1	1	1	1	2	1	2	2	1	1	1	1	1	2	1	29			
	Participant 13	2	1	2	2	2	1	1	2	1	4	1	3	2	1	1	1	1	2	2	2	2	1	3	1	2	2	2	32			
	Participant 14	2	1	2	2	2	1	1	2	1	1	1	2	2	1	1	1	1	2	1	2	1	2	1	1	1	1	1	27			
	Participant 15	1	3	1	1	1	3	1	1	1	1	1	1	1	1	1	1	2	1	1	2	3	1	2	2	1	2	3	24			
	Participant 16	1	2	1	1	1	4	1	1	2	1	1	1	1	2	2	2	1	1	2	1	1	2	2	1	1	1	1	28			
±3 Years (N=12)	Participant 17	1	2	1	1	1	2	2	1	1	4	2	1	1	1	1	1	2	1	3	1	2	1	2	2	1	1	4	29	1.62	1.59	3.21
	Participant 18	4	1	4	1	4	1	1	4	1	1	1	4	4	1	2	4	2	4	3	2	[illegible]	1	1	2	3	1	1	47			
	Participant 19	1	1	1	4	1	1	2	1	1	2	1	1	1	3	2	1	1	1	1	2	2	1	1	1	1	1	2	27			
	Participant 20	1	1	1	1	1	1	2	1	1	2	2	1	1	1	1	1	1	1	2	3	1	1	1	3	1	2	2	23			
	Participant 21	1	1	1	1	1	1	1	1	2	3	1	1	3	2	1	2	1	1	1	1	1	1	1	3	3	1	1	26			
	Participant 22	1	4	1	1	1	2	1	1	1	1	1	1	1	1	1	1	1	1	2	2	1	1	1	2	2	1	2	24			
	Participant 23	1	1	1	2	1	1	1	1	2	2	1	1	1	2	1	2	2	1	2	2	2	2	2	2	1	1	2	26			
	Participant 24	2	1	2	2	2	1	1	2	2	4	4	2	2	2	2	2	1	2	1	2	1	2	2	2	2	1	1	37			
	Participant 25	2	1	2	1	2	1	2	2	1	2	1	2	2	1	1	1	2	2	1	3	1	1	1	1	3	2	1	29			
	Participant 26	3	2	3	1	3	1	1	3	1	3	1	1	3	1	2	1	1	3	1	1	1	1	3	1	2	2	1	35			
	Participant 27	3	2	3	2	3	2	2	3	1	1	1	1	3	1	2	1	1	3	2	1	1	3	1	1	2	3	2	37			
	Participant 28	1	1	1	3	1	1	4	1	1	1	1	4	1	1	1	1	1	1	1	2	1	2	1	1	1	2	3	27			

±4 Years (N=7)	Participant 29	2	1	2	1	2	2	2	2	2	2	1	1	2	2	1	2	1	2	1	1	1	1	2	1	1	1	1	31	1.70	1.64	3.34
	Participant 30	4	1	4	3	1	2	3	1	4	1	2	1	1	1	2	1	4	4	1	1	1	1	1	1	1	3	1	41			
	Participant 31	2	1	2	3	2	1	1	2	1	1	2	1	2	1	3	1	2	2	1	1	1	1	1	1	1	3	3	31			
	Participant 32	3	1	3	1	3	1	1	3	2	2	3	1	3	2	3	2	2	3	2	4	2	4	1	1	1	2	2	41			
	Participant 33	1	1	1	2	1	1	2	1	1	1	3	1	1	1	2	1	2	1	1	1	2	1	4	2	2	2	1	25			
	Participant 34	1	1	1	1	1	1	1	1	2	1	2	2	1	2	1	2	1	1	4	1	1	1	1	2	2	2	2	27			
	Participant 35	2	2	2	2	2	2	1	2	2	1	1	2	2	2	1	2	2	2	1	1	1	1	1	1	3	1	3	33			
±5 Years (N=4)	Participant 36	1	1	1	3	1	4	3	1	1	1	1	3	1	1	1	1	2	1	1	1	1	2	1	1	3	1	2	29	1.78	1.75	3.53
	Participant 37	1	1	1	1	1	2	2	1	1	4	4	3	4	1	1	1	4	1	1	4	1	2	3	2	2	2	2	35			
	Participant 38	3	1	3	1	3	3	1	3	1	1	1	3	3	1	2	1	1	3	1	2	2	2	2	2	3	1	3	36			
	Participant 39	2	1	2	2	2	1	2	2	1	1	2	3	2	1	1	1	2	2	1	2	1	1	3	1	3	1	1	31			
±6 Years (N=6)	Participant 40	1	2	1	1	1	1	3	4	2	1	1	2	1	2	2	2	3	1	2	3	1	1	3	1	1	1	2	33	1.84	1.82	3.66
	Participant 41	1	2	1	1	4	2	2	1	1	1	1	4	4	1	1	4	3	3	2	1	2	3	1	1	1	1	2	39			
	Participant 42	1	1	4	2	4	1	2	4	1	1	2	1	4	1	1	1	2	1	3	1	2	2	1	1	2	1	3	37			
	Participant 43	1	1	4	2	4	1	1	4	2	2	1	4	1	2	1	2	1	4	1	2	1	4	2	1	1	1	3	39			
	Participant 44	1	1	1	1	1	2	1	1	1	2	4	1	1	1	2	1	4	1	1	2	4	1	1	1	1	1	1	28			
	Participant 45	3	2	3	1	3	1	1	3	1	3	3	2	3	1	2	1	2	3	1	3	1	3	2	1	3	2	2	39			
																															Total Mean of Post-Control	3.66
																															Std. Daviation	0.22

Appendix D

RAW DATA OF PRE-TEST SCORES

Experimental Group (N=34)

		Familiar Content															Unseen Contents												Result			
Experience	Participants	Q 1.1	Q 1.2	Q 1.3	Q 1.4	Q 1.5	Q 1.6	Q 1.7	Q 2.1	Q 2.2	Q 2.3	Q 2.4	Q 3.1	Q 3.2	Q 3.3	Q 3.4	Q 3.5	Q 3.6	Q 3.7	Q 3.8	Q 3.9	Q 3.10	Q 4.1	Q 4.2	Q 4.3	Q 4.4	Q 4.5	Q 4.6	Total	Mean of Familiar	Mean of Unseen	Total Mean
±2 Years (N=8)	Participant 1	4	1	1	1	3	2	1	1	3	2	1	1	2	1	1	2	1	2	1	2	1	1	3	2	1	3	2	31	1.55	1.53	3.08
	Participant 2	1	1	1	2	1	1	1	1	1	2	2	1	1	2	2	1	1	1	2	1	1	1	2	2	2	1	1	25			
	Participant 3	1	3	1	3	2	1	1	2	1	3	1	2	1	1	1	2	2	1	1	1	1	2	1	3	1	2	1	30			
	Participant 4	2	2	2	1	1	1	1	1	1	2	4	1	1	1	1	1	1	2	1	1	1	1	2	1	1	1	2	27			
	Participant 5	2	2	2	1	1	1	1	1	1	1	2	1	1	1	1	4	1	1	2	1	1	1	2	2	1	1	3	27			
	Participant 6	4	1	4	2	1	1	1	1	1	2	2	3	4	1	1	1	1	1	2	1	1	1	2	2	3	2	1	34			
	Participant 7	1	2	3	2	1	1	2	2	1	3	1	1	1	2	1	3	1	2	2	2	2	1	3	1	1	1	1	32			
	Participant 8	2	1	3	1	1	1	1	2	1	1	3	1	2	2	1	1	1	2	1	1	2	1	1	3	1	2	3	28			
±3 Years (N=7)	Participant 9	3	1	1	1	2	2	1	2	1	2	3	3	1	1	1	1	2	1	1	1	2	1	1	1	1	2	1	30	1.57	1.56	3.13
	Participant 10	4	1	1	1	2	1	2	2	1	1	1	2	1	2	1	4	1	2	2	1	1	2	2	1	2	3	1	32			
	Participant 11	1	1	1	2	1	2	4	1	2	1	1	1	4	1	2	2	1	1	1	2	2	3	1	2	2	1	2	30			
	Participant 12	1	3	1	3	2	3	2	2	1	1	2	4	2	1	1	1	2	1	2	1	1	2	1	2	1	2	2	35			
	Participant 13	2	2	2	1	1	1	1	2	1	4	1	1	1	1	1	1	1	2	2	2	1	2	2	1	2	1	3	28			
	Participant 14	2	2	2	1	1	1	1	2	1	1	4	1	1	1	1	1	1	2	1	2	2	1	1	2	1	1	2	29			
	Participant 15	1	3	1	1	1	3	1	1	1	1	1	1	1	1	1	1	2	1	1	3	2	2	1	1	2	1	2	24			
±4 Years (N=8)	Participant 16	1	2	4	1	1	3	1	1	2	1	1	1	1	2	2	2	2	1	2	2	2	2	1	4	1	2	1	31	1.62	1 60	3.22
	Participant 17	1	2	1	1	1	2	2	1	1	1	2	1	2	1	4	1	1	4	1	2	1	1	4	2	1	1	1	30			
	Participant 18	4	1	3	1	3	2	1	3	1	1	1	1	1	3	1	1	1	1	2	1	1	2	1	1	2	1	1	32			
	Participant 19	1	4	1	4	1	1	2	1	1	2	1	2	1	1	1	1	2	4	2	2	1	2	2	1	1	2	1	33			
	Participant 20	1	4	4	1	1	1	2	2	1	2	2	4	1	1	1	1	2	1	2	2	2	1	4	2	1	1	4	34			
	Participant 21	1	1	1	1	1	1	1	2	2	3	1	2	1	1	1	2	1	2	2	2	2	1	1	1	1	1	1	27			
	Participant 22	1	4	4	1	1	2	1	2	1	1	1	1	1	1	1	1	1	2	1	1	3	2	1	2	1	2	1	28			
	Participant 23	1	1	4	2	1	1	4	1	2	2	1	1	1	2	1	2	2	1	1	2	1	2	2	1	2	1	1	31			

±5 Years (N=6)	Participant 24	2	1	2	4	2	1	1	2	3	2	2	2	2	2	2	2	1	1	2	2	2	1	3	2	1	1	1	**36**	1.67	1.64	3.31
	Participant 25	1	1	2	1	2	2	2	2	1	1	1	2	2	1	1	1	3	2	2	3	2	2	1	1	2	2	1	**30**			
	Participant 26	1	2	1	1	3	1	1	1	1	2	1	1	3	4	2	1	1	4	2	1	2	2	1	2	1	2	2	**33**			
	Participant 27	3	2	3	2	1	2	2	2	1	1	1	1	3	1	2	2	1	3	2	2	1	1	2	1	2	1	3	**35**			
	Participant 28	1	1	1	3	1	1	4	1	1	1	1	2	1	1	1	1	1	1	1	1	2	2	3	2	1	2	1	**25**			
	Participant 29	2	1	2	1	2	2	2	2	2	2	1	1	2	2	1	2	1	2	1	2	1	2	1	1	1	2	1	**31**			
±6 Years (N=5)	Participant 30	4	1	2	3	2	2	3	2	1	1	2	1	2	1	2	2	1	2	1	1	2	1	1	2	1	1	2	**35**	1.72	1.67	3.39
	Participant 31	2	1	2	1	2	1	1	2	1	1	2	2	4	3	1	2	2	2	2	4	1	2	2	2	2	1	1	**34**			
	Participant 32	3	1	3	2	2	1	1	1	2	2	1	2	2	3	2	4	1	3	2	1	2	2	1	2	1	1	1	**38**			
	Participant 33	1	1	1	2	1	1	2	1	1	1	2	1	3	1	2	1	2	2	1	2	2	1	1	1	2	2	1	**27**			
	Participant 34	1	1	1	4	1	1	1	3	2	2	2	2	1	2	1	2	1	1	3	2	1	2	1	2	3	1	2	**32**			
																															Total Mean of Pre-Experimental	3.23
																															Std. Daviation	0.12

Appendix E

RAW DATA OF POST-TEST SCORES

Control Group (N=45)

		Familiar Content															Unseen Contents												Result		
Experience	Participants	Q 1.1	Q 1.2	Q 1.3	Q 1.4	Q 1.5	Q 1.6	Q 1.7	Q 2.1	Q 2.2	Q 2.3	Q 2.4	Q 3.1	Q 3.2	Q 3.3	Q 3.4	Q 3.5	Q 3.6	Q 3.7	Q 3.8	Q 3.9	Q 3.10	Q 4.1	Q 4.2	Q 4.3	Q 4.4	Q 4.5	Q 4.6	Total	Mean of Familiar	Mean of Unseen	Total Mean
±2 Years (N=16)	Participant 1	5	2	3	1	5	1	2	5	5	2	3	5	5	5	5	5	5	1	1	3	5	5	5	2	5	5	2	66	3.05	2.95	6.00
	Participant 2	2	5	5	5	5	5	5	5	3	2	2	1	5	1	2	1	1	1	2	4	1	5	1	4	1	1	2	58			
	Participant 3	1	5	5	5	2	5	1	5	5	2	5	2	2	2	5	5	5	3	1	5	1	1	4	1	3	2	5	66			
	Participant 4	1	2	2	5	3	1	5	5	2	2	5	1	1	1	5	5	1	3	2	5	1	3	1	3	1	2	5	52			
	Participant 5	3	5	5	5	2	3	5	1	1	2	5	1	1	4	1	1	1	2	1	5	4	5	5	5	2	5	5	49			
	Participant 6	4	1	5	2	1	2	5	5	5	1	2	3	5	5	5	5	5	5	1	5	1	1	2	5	5	5	5	67			
	Participant 7	1	2	5	2	1	1	2	5	5	3	5	1	5	2	1	3	1	5	2	5	4	1	2	5	3	5	4	52			
	Participant 8	2	5	3	1	5	5	5	2	5	1	3	1	2	2	1	1	1	5	1	5	1	3	2	4	3	4	2	51			
	Participant 9	3	5	1	1	2	5	5	5	5	2	5	5	5	1	1	1	2	1	1	5	2	5	4	5	5	5	2	56			
	Participant 10	1	5	5	1	5	5	5	2	1	1	2	2	5	2	5	1	1	2	2	1	5	1	1	1	2	5	5	53			
	Participant 11	5	2	1	1	1	2	1	1	2	1	2	5	5	5	2	2	1	1	1	2	3	5	4	5	2	5	3	41			
	Participant 12	1	2	1	3	1	3	2	1	1	1	2	5	1	5	1	1	2	1	2	3	1	5	1	3	2	4	3	36			
	Participant 13	5	5	2	2	2	5	1	2	1	4	5	5	5	5	5	1	1	2	2	5	2	5	2	5	5	5	2	60			
	Participant 14	2	1	2	2	2	1	5	2	1	5	5	5	2	5	5	1	1	2	1	5	5	1	5	2	2	5	3	50			
	Participant 15	5	5	1	5	5	5	5	1	1	5	5	5	5	1	5	1	2	1	1	3	5	2	2	5	5	5	2	64			
	Participant 16	1	2	1	1	1	4	1	1	2	1	5	1	5	2	2	2	1	1	2	5	2	5	5	5	4	5	5	36			
±3 Years (N=12)	Participant 17	1	2	5	5	1	2	2	1	1	1	2	1	1	1	1	1	2	1	3	5	4	3	4	2	2	2	5	34	3.26	3.07	6.33
	Participant 18	4	1	5	1	5	1	1	5	1	1	1	4	4	5	2	1	2	4	3	4	2	3	2	1	5	5	4	51			
	Participant 19	1	5	5	5	5	5	2	5	1	5	1	5	5	5	2	1	1	1	2	1	3	1	3	3	5	2	5	62			
	Participant 20	5	5	1	1	1	5	2	5	1	5	5	5	5	5	1	1	1	1	1	5	1	2	5	1	5	2	5	56			
	Participant 21	5	5	5	5	5	5	1	1	2	3	5	5	5	5	1	2	5	5	2	1	5	5	5	5	5	5	3	72			
	Participant 22	1	5	1	5	1	5	1	5	5	1	1	5	5	5	1	5	1	1	1	1	4	5	5	5	2	5	5	55			
	Participant 23	1	5	5	5	5	3	1	4	5	2	5	5	5	5	5	2	2	5	1	1	5	5	2	5	5	5	5	71			
	Participant 24	5	5	5	5	2	5	1	2	2	4	4	2	5	2	2	2	1	2	2	5	4	5	2	5	3	5	4	58			
	Participant 25	5	1	5	5	5	5	2	5	5	5	1	5	2	5	1	1	4	2	1	5	1	3	2	4	3	4	2	65			
	Participant 26	3	2	5	5	3	5	1	5	5	5	5	5	3	1	2	5	1	3	2	5	2	5	5	5	2	5	3	66			
	Participant 27	5	2	3	2	5	2	2	3	1	1	1	1	3	1	5	1	1	5	2	1	5	2	2	5	3	5	5	46			
	Participant 28	1	5	1	3	5	1	4	1	1	5	5	5	5	1	1	1	5	1	1	1	2	1	5	5	2	5	2	52			
±4 Years (N=7)	Participant 29	5	5	5	5	2	5	2	5	2	5	1	5	2	5	1	2	1	2	1	5	5	5	4	5	5	5	5	61	3.47	3.13	6.60
	Participant 30	4	1	4	3	5	5	3	5	5	1	2	2	4	5	2	1	1	4	1	5	5	5	4	1	5	2	5	58			
	Participant 31	5	5	2	3	2	5	5	2	5	5	2	2	2	5	5	2	2	2	1	3	2	2	3	1	4	5	4	62			
	Participant 32	5	1	3	2	3	2	2	3	2	5	3	2	1	2	5	1	2	3	5	3	2	2	5	3	3	3	5	52			
	Participant 33	1	2	2	2	5	2	2	5	1	5	3	1	5	5	2	2	2	1	1	5	3	5	5	2	4	5	3	49			

	Participant 34	2	5	1	5	5	2	5	5	5	5	5	5	2	2	5	2	1	1	5	1	3	5	5	3	3	5	5	68			
	Participant 35	5	2	5	5	5	2	5	5	2	5	5	2	5	2	5	2	2	2	1	5	1	4	5	3	4	3	2	67			
±5 Years (N=4)	Partic.pant 36	1	1	5	3	5	4	3	5	1	5	5	5	5	5	1	5	2	1	5	2	1	4	15	1	2	1	5	67	3.75	3.21	6.96
	Participant 37	5	5	5	5	5	5	5	5	5	4	5	3	5	5	5	5	5	5	1	2	2	3	5	1	1	2	2	88			
	Participant 38	5	5	3	5	5	3	5	3	5	5	5	3	3	1	5	1	1	3	1	5	2	5	3	2	5	5	1	67			
	Participant 39	2	1	5	2	2	1	5	5	1	1	2	3	2	5	1	4	5	5	5	5	1	5	5	2	1	3	1	57			
±6 Years (N=6)	Participant 40	5	5	5	5	4	4	3	5	4	5	5	2	5	2	2	2	3	1	2	3	3	1	3	2	4	3	4	69	4.03	3.31	7.34
	Participant 41	5	2	5	5	5	5	5	1	5	5	4	1	5	5	5	5	3	1	5	4	4	2	5	5	5	2	5	77			
	Participant 42	5	5	5	5	4	5	2	5	5	1	2	5	5	5	5	1	5	1	3	3	5	5	2	2	5	3	5	74			
	Participant 43	5	5	5	5	5	5	1	5	5	5	1	5	4	5	1	5	1	5	1	2	3	2	1	5	5	2	5	74			
	Participant 44	5	5	5	5	1	5	5	5	1	5	5	5	5	5	5	1	1	1	5	4	4	5	5	4	5	5	5	75			
	Participant 45	5	2	3	1	3	1	5	3	5	3	3	2	3	5	2	1	2	3	1	3	3	5	5	4	5	5	2	53			
																															Total Mean of Post-Control	7.34
																															Std. Daviation	0.52

Appendix F

RAW DATA OF POST-TEST SCORES

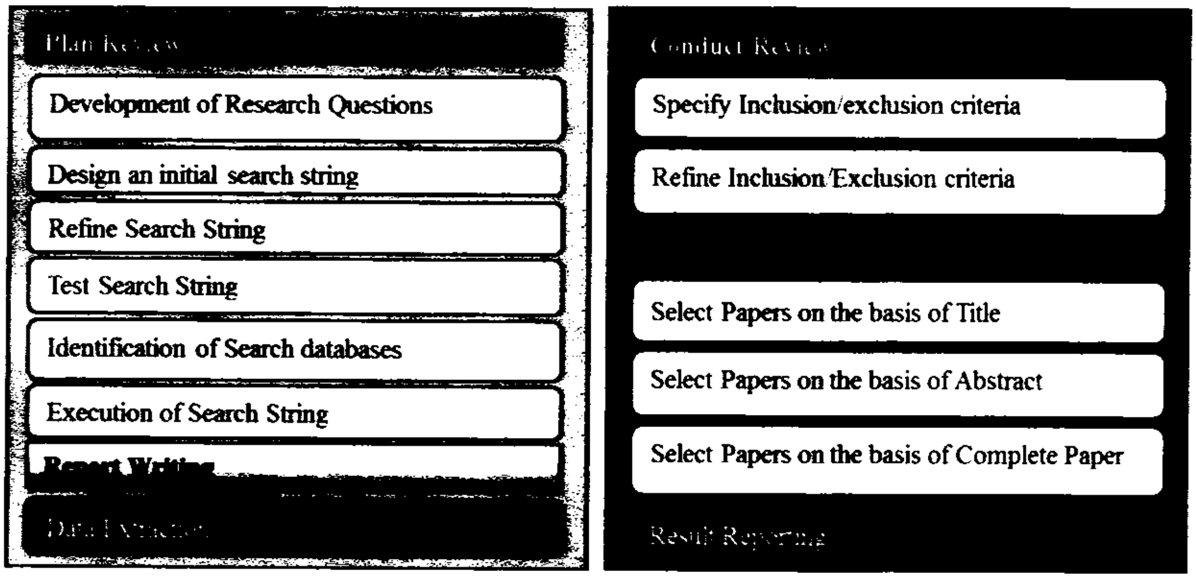
Experimental Group (N=34)

		Familiar Content															Unseen Contents												Result		
Experience	Participants	Q 1.1	Q 1.2	Q 1.3	Q 1.4	Q 1.5	Q 1.6	Q 1.7	Q 2.1	Q 2.2	Q 2.3	Q 2.4	Q 3.1	Q 3.2	Q 3.3	Q 3.4	Q 3.5	Q 3.6	Q 3.7	Q 3.8	Q 3.9	Q 3.10	Q 4.1	Q 4.2	Q 4.3	Q 4.4	Q 4.5	Q 4.6	Total	Mean of Familiar	Mean of Unseen	Total Mean
±2 Years (N=8)	Participant 1	4	5	5	5	3	2	5	5	3	2	5	3	2	5	5	2	5	3	3	3	5	1	1	2	5	5	2	72	3.89	3.50	7.39
	Participant 2	5	2	5	3	5	5	4	5	4	2	2	3	5	4	4	5	5	5	2	4	1	5	1	4	1	1	2	75			
	Participant 3	5	1	5	2	3	5	5	4	5	5	5	2	5	2	5	3	5	5	5	5	5	5	4	1	3	2	5	77			
	Participant 4	2	5	2	5	2	5	5	5	5	2	5	5	5	5	5	2	3	5	5	5	5	3	1	3	1	2	5	78			
	Participant 5	2	5	2	5	3	2	5	5	5	3	2	5	5	5	2	5	2	5	2	2	5	5	5	3	5	1	2	70			
	Participant 6	4	5	4	2	5	4	5	5	5	2	2	3	4	5	2	3	5	2	2	4	5	4	2	5	4	5	5	69			
	Participant 7	5	2	3	2	5	5	2	2	5	3	5	5	5	2	5	3	5	2	2	5	2	3	2	5	5	2	2	68			
	Participant 8	2	5	3	5	5	5	5	2	5	5	3	5	2	2	5	5	5	2	5	2	5	3	5	5	5	5	2	76			
±3 Years (N=7)	Participant 9	3	5	5	5	2	5	5	2	5	2	3	3	5	5	5	5	2	5	5	2	5	1	4	5	4	5	2	77	3.96	3.54	7.50
	Participant 10	4	1	5	5	4	1	5	2	5	5	5	5	5	5	4	5	5	2	2	4	3	5	5	5	1	3	2	75			
	Participant 11	5	5	1	4	5	3	2	5	2	5	5	5	5	4	2	3	3	5	5	2	3	5	3	5	2	5	5	74			
	Participant 12	5	1	3	1	3	1	2	5	5	5	5	4	2	2	2	3	2	5	2	3	1	5	1	1	1	2	5	58			
	Participant 13	2	5	5	5	3	5	5	2	5	4	5	5	5	5	3	5	3	2	2	5	2	4	2	5	2	1	5	76			
	Participant 14	2	5	1	5	1	5	5	2	5	5	4	5	5	5	1	5	5	2	5	5	5	5	1	1	5	5	5	73			
	Participant 15	5	3	5	5	5	3	5	5	5	5	5	5	5	5	5	5	2	5	5	3	5	2	2	5	5	5	2	88			
±4 Years (N=8)	Participant 16	5	2	5	5	5	4	5	5	2	5	5	5	5	2	2	2	5	5	2	5	2	1	5	5	4	5	5	76	4.01	3.43	7.44
	Participant 17	5	4	3	4	2	2	2	5	5	5	2	5	4	5	4	5	2	5	3	1	4	3	4	2	2	2	5	72			
	Participant 18	4	2	3	2	5	5	5	4	5	5	5	2	1	5	3	3	2	4	3	4	2	3	2	1	5	5	4	68			
	Participant 19	5	3	5	3	3	5	2	5	5	2	5	2	5	5	2	3	5	5	2	5	3	1	3	3	1	2	5	72			
	Participant 20	5	2	2	2	5	5	2	2	5	2	2	5	5	5	5	2	5	5	5	5	5	2	5	1	1	2	5	71			
	Participant 21	5	5	5	5	5	5	5	3	2	3	5	4	2	5	2	1	1	5	2	5	5	5	5	1	5	1	3	70			
	Participant 22	5	4	5	5	5	2	5	5	5	5	5	5	5	5	5	5	1	5	5	5	4	1	5	1	2	5	5	87			

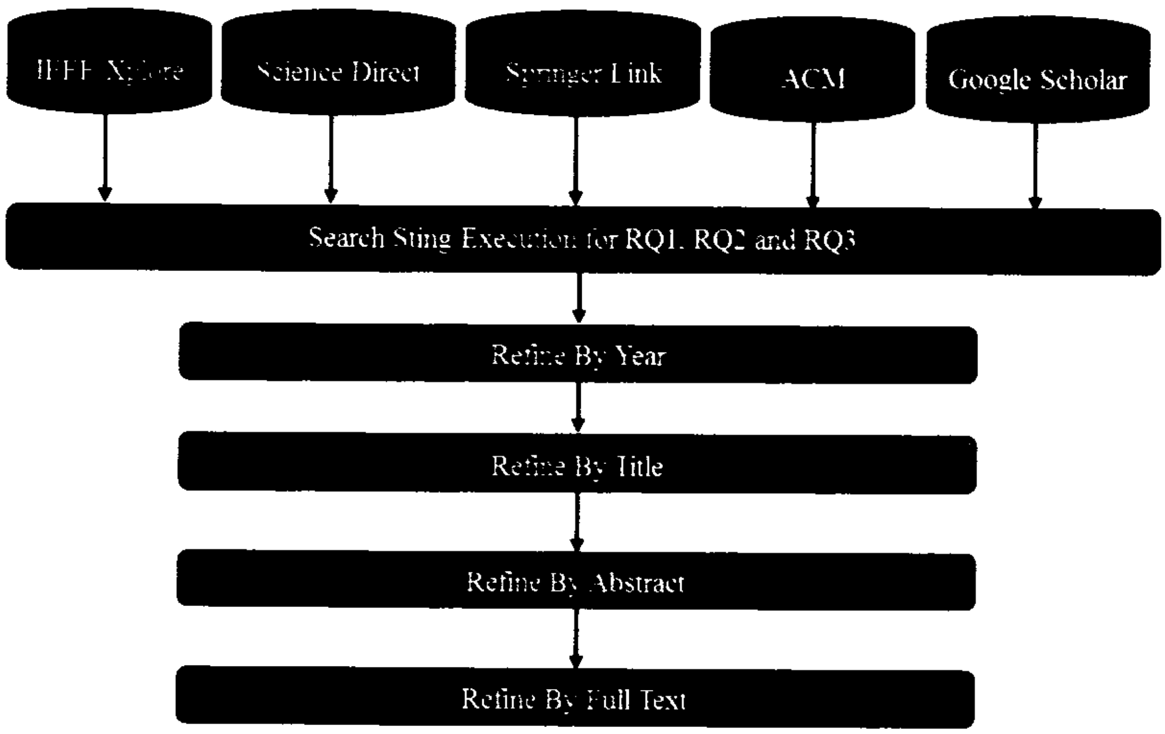
	Participant 23	5	5	5	2	5	5	5	5	2	2	5	5	5	2	5	2	2	1	5	5	5	1	2	5	5	5	5	73			
±5 Years (N=6)	Participant 24	2	5	2	5	5	5	5	5	5	4	4	5	5	5	5	2	5	2	5	5	4	5	2	5	3	5	4	81	4.14	3.85	7.99
	Participant 25	2	5	2	5	4	5	5	4	5	5	5	5	5	5	5	5	4	5	5	5	4	4	2	4	3	4	2	86			
	Participant 26	3	3	3	5	3	5	4	3	5	3	5	5	5	4	5	3	5	3	2	5	2	1	1	5	2	5	3	74			
	Participant 27	3	4	3	2	5	2	2	3	5	5	5	4	5	5	4	5	3	3	2	4	5	2	2	5	3	5	5	70			
	Participant 28	5	4	5	3	5	5	4	5	5	5	5	4	5	5	5	5	5	5	5	5	2	3	5	5	2	5	2	90			
	Participant 29	2	5	2	5	2	2	2	2	5	2	5	5	2	5	5	2	5	2	5	5	5	5	4	5	5	5	5	65			
±6 Years (N=5)	Participant 30	4	5	4	3	4	4	3	4	5	5	2	5	4	5	2	5	5	4	5	2	5	2	4	5	5	2	5	78	4.25	3.98	8.24
	Participant 31	5	5	5	3	5	5	5	5	5	5	2	2	5	3	5	2	5	5	2	3	5	5	3	5	3	5	4	79			
	Participant 32	3	5	3	3	3	5	5	3	2	2	5	2	5	4	3	5	2	4	2	3	5	2	2	3	4	3	5	68			
	Participant 33	5	5	5	5	5	5	5	5	5	5	2	5	5	5	2	5	3	5	5	5	5	5	5	5	3	5	3	87			
	Participant 34	5	5	5	4	5	5	5	5	5	5	5	5	5	2	5	2	5	5	4	5	3	5	5	5	3	5	2	87			
																															Total Mean of Post-Experimental	7.71
																															Std. Daviation	0.38

Aappendix G

Protocol of the Systematic Mapping Study Process:


Plan Review
Development of Research Questions
Design an initial search string
Refine Search String
Test Search String
Identification of Search databases
Execution of Search String
Report Writing
Data Extraction
Conduct Review
Specify Inclusion/exclusion criteria
Refine Inclusion/Exclusion criteria
Select Papers on the basis of Title
Select Papers on the basis of Abstract
Select Papers on the basis of Complete Paper
Result Reporting

Process of Finalizing the Search String


IEEE Xplore
Science Direct
Springer Link
ACM
Google Scholar
Search Sting Execution for RQ1, RQ2 and RQ3
Refine By Year
Refine By Title
Refine By Abstract
Refine By Full Text