SOCIAL CONDITIONS LEADING TO SCRUM PROCESS BREAKDOWNS DURING GLOBAL AGILE SOFTWARE DEVELOPMENT: A THEORY OF PRACTICE PERSPECTIVE

****

Thesis Presented for the Degree of

DOCTOR OF PHILOSOPHY

In the Department of Information Systems

UNIVERSITY OF CAPE TOWN

November 2013

By:

Maureen Cynthia TANNER

TNNMAU001

DEDICATION

To Dad, Mr Georges Steve Tanner

Thank You Papi!

Your love, support and faith in me have brought me to where I am today

I miss you everyday

ACKNOWLEDGEMENTS

Thank you to my supervisor, **Professor Wallace Chigona** of the University of Cape Town. His guidance, constant encouragement, thoughts and advice were invaluable to me throughout this journey.

I am indebted as well to **Professor Kosheek Sewchurran** of the University of Cape Town. He has been a strong source of inspiration and his comments and advice largely contributed to this thesis.

My deepest thanks also go to **Professor Ojelanki Ngwenyama** for his invaluable guidance, advice, and support.

To all the participants of this study who generously devoted their time and welcomed me in their country and work environment. This PhD work would never have been completed without your input and I am forever grateful for your contribution.

I would also like to thank all my colleagues and friends at the University of Cape Town. Thank you for all these amazing opportunities which have been given to me during the past four years, which allowed me to complete this thesis.

Last but not least, I would like to express my utmost gratitude to my mother and brother. Their patience, understanding, and constant encouragement supported me throughout this long journey. I would never have completed this thesis without their faith in me.

DECLARATION

STUDENT NAME: MAUREEN C TANNER
STUDENT NUMBER: TNNMAU001

1. I know that plagiarism is wrong. Plagiarism is to use another's work and pretend that it is one's own.

2. I have used the APA convention for citation and referencing. Each contribution to, and quotation in, this PHD thesis, from the work(s) of other people has been attributed, and has been cited and referenced.

3. This **PhD Thesis** is my own work.

4. I have not allowed, and will not allow, anyone to copy my work with the intention of passing it off as his or her own work.

5. I acknowledge that copying someone else's assignment or essay, or part of it, is wrong, and declare that this is my own work.

SIGNATURE: Maureen C TANNER
DATE: April 2013

ABSTRACT

Global Software Development (GSD) and Agile are two popular software development trends that are gaining in popularity. In addition, more and more organisations are now seeking to engage in agile software development within the GSD context to reap the benefits of both ventures and achieve project success. Hence, agile methodologies adapted to fit the GSD context are commonly termed Global Agile Software Development (GASD) methodologies. However, because of geographical, temporal, and cultural challenges, collaboration is not easily realized in the GASD context. In addition, this work context is characterized by multiple overlapping fields of practice, which further hinder collaboration, and give rise to social challenges. Given the existence of these social challenges, there is a need to further investigate the human-centred aspect of collaboration during GASD.

Following an extensive literature review on the application of Scrum and other agile methodologies in GASD between 2006 and 2011, it was noted that there is a lack of understanding of the social conditions giving rise to the social challenges experienced during GASD. It was noted that past studies have instead sought to describe these social challenges and to provide mitigating strategies in the form of best-practices, without detailing and theorising about the social conditions under which these social challenges emerge.

One of the objective of the study was thus to investigate the use of Scrum during GASD. In particular, the Scrum process breakdowns experienced during and after Scrum's sprint planning and retrospective meetings were identified. The social conditions under which these breakdowns emerged were investigated in the light of Bourdieu's Theory of Practice. Scrum Process breakdowns were defined as *any deviation from an ideal Scrum process (as per the Scrum methodology's guidelines) which yields to the emergence of social challenges, conflict or disagreements in the GASD team.*

The study was empirical and qualitative in nature and followed the positivist research paradigm. Two case studies, in line with Bonoma (1985)'s "drift" and "design" stages of case study design, were undertaken to investigate the phenomena of interest and answer the research questions. The first case focused on a distributed agile team executing a

software project across South Africa (Cape Town) and Brazil (Sao Paulo) while the second case focused on a team executing an agile software project across India (Pune) and South Africa (Durban). The site selection was carefully thought out and the results from the first case informed the second case in order to add more richness in the data being gathered. In both case studies, data was collected through semi-structured interviews, documentation, field notes and direct observation. The underlying theoretical framework employed for the study was the Theory of Practice (Bourdieu, 1990).

The study has identified various forms of Scrum process breakdowns occurring during and after sprint planning and retrospective meetings:

- Different perceptions about task urgency at the software development sites
- Disagreements on the suitability of software engineering practices
- Low level of communication openness during meetings involving the whole GASD team compared to internal meetings at the sites
- Impromptu changes to user stories' content and priorities
- Product Owner's low level of authority
- Disagreements on estimation mechanisms
- Number of User Stories to be completed during the Sprint is imposed on the team
- Decisions on Scrum process updates not made by the development team
- Selective invitation to retrospective meetings

In addition, various social conditions were identified as possibly leading to the emergence of these Scrum process breakdowns in the GASD context:

- GASD project stakeholders' low level of capital in the joint field
- Different beliefs and values because of multiple fields

Two theoretical propositions were derived to describe the social conditions and the corresponding Scrum process breakdowns which are likely to emerge under these conditions.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ACRONYMS

BPO - Business Process Outsourcing

CEO – Chief Executive Officer

CiTi – Cape IT Innitiative

CMM - Capability Maturity Model

COO – Chief Operations Officer

GSD – Global Software Development

GASD – Global Agile Software Development

IS – Information Systems

MVC – Model View Controller

QA – Quality Assurance

SA – South Africa

TP – Theoretical Proposition

UK – United Kingdom

UML – Unified Modelling Language

US – United States

WSF – Work Systems Framework

XP – eXtreme Programming

1. INTRODUCTION

1.1 Background to Research Problem

Software development practices are constantly evolving and Global Software Development (GSD) is among one of the growing trends (Aspray, Mayadas, & Vardi, 2006). GSD can be defined as a network of interactions between people, organisation and technology, engaged in the development of software projects across various countries. It involves a variety of human and social issues, such as relationships of people, teams, organisations, and nations with different backgrounds, languages and working styles (Herbsleb & Mockus, 2003). GSD can be implemented through both outsourcing and distributed teams spread across different countries (Herbsleb & Mockus, 2003). Organisations embracing this concept aim for cost reductions, access to more sophisticated skills and labour markets, improved operational performance, flexibility and efficiency of operation, as well as strategic advantage in the marketplace (Aron & Singh, 2005; Dibbern, Goles, Hirschheim, & Jayatilaka, 2004; Krishna, Sahay, & Walsham, 2004; Tas & Sunder, 2004; Gopal, Mukhopadhyay, & Krishman, 2002).

Another current trend in the software development industry is the adoption of light-weight and agile software development methodologies as espoused by the Agile Alliance. The aim of agile software development methodologies is to limit risks by developing software in short time-boxes called iterations. These short iterations typically last one to four weeks at the end of which, potential shippable increments are. (Agile Alliance, 2001). Furthermore, by emphasising the importance of individuals and interactions over processes and tools, working software over comprehensive documentation, customer collaboration over contract negotiation, and responding to change over following a plan (Agile Alliance, 2001), agile software development strives to overcome common challenges to software development (e.g. excessive time taken to understand the user requirements, user requirements not being met, inappropriate software being developed) (Abrahamson, Salo, Ronkainen, & Warsta, 2002). According to agile precepts, members of an agile team should be empowered and trust each other, should acknowledge change and promote constant feedback (Schuh, 2005, p. 2).

Agile principles, which are at the basis of the various agile software development methodologies, have emerged in response to constantly changing and volatile business requirements (Sone, 2008), which is also a recurring phenomenon in GSD (Damian &

Zowghi, 2003). It is therefore not surprising that many GSD organisations are turning to agile methodologies to address GSD challenges (VersionOne, 2012). However, due to temporal, geographical, and socio-cultural distance prevailing in GSD (Herbsleb & Moitra, 2001), it is challenging to realise agile methodologies in that context (Maurer & Martel, 2002; Kirscher, Jain, Corsaro, & Levine, 2001). Such is the case since agile methodologies rely on high levels of interactions among team members (Agile Alliance, 2001) and these interactions are mostly realised via informal communication which is not achieved in GSD. (Ibarra & Andrews, 1993).

Informal communication relates to spontaneous, interactive and rich forms of interpersonal communication. Whilst not relying on any pre-specification, information is often exchanged interactively through meetings and "corridor" conversations (Kraut, Fish, Root, & Chalfonte, 1990). Informal communication can be perceived as being more effective than formal channels, as participants in the conversation elaborate or modify what they have to say in order to deal with someone else's objections or misunderstandings (e.g. Kraut, Lewis, & Swezey, 1982). Through these informal encounters, people get to know each other, thus creating a common context and perspective to support planning and coordination in group work (Kraut et al., 1990).

Some researchers (e.g. Conboy & Fitzgerald, 2004; Maurer & Martel, 2002) argue that the use of agile software development methodologies might even mitigate the negative impact of distance during GSD. These agile methodologies, adapted to fit the GSD context, are commonly termed Global Agile Software Development (GASD) methodologies (e.g. (Ngo-The, Hoang, Nguyen, & Mai, 2005; Kirscher et al., 2001; Repenning, Ioannidou, Payton, Wenming, & Roschelle, 2001).

1.2 Problem Statement

The use of agile methodologies in the GASD setting is difficult to realise (Paasivaara, Durasiewicz, & Lassenius, 2008). Based on agile principles, participants are expected to collaborate but collaboration under such conditions is harder to realise because of multiple and overlapping fields of practice related to the multi-country context (Levina & Vaast, 2008; Hinds & Bailey, 2003).

As is the case for any large-scale software development projects, collaboration in GASD involves team members engaged in joint work (Levina & Vaast, 2008) and occurs at various levels. For instance Scrum, an agile software development methodology often

employed in the GASD context, requires participants to collaborate while they engage in various types of activities. This study particularly focuses on two Scrum activities: sprint planning meetings and retrospective meetings. Breakdowns experienced when GASD team members collaborate during these activities, or when decisions made during these activities are overruled during the sprint, are also investigated. The study posits that breakdowns occur when the process followed by the GASD team deviates from an ideal Scrum (as specified by the Scrum methodology), thus leading to social challenges. The Oxford Dictionary defines a breakdown as "a failure of a relationship or system" (OxfordDictionary, 2013). In this study, these breakdowns have been specifically termed as *Scrum Process Breakdowns.* In light of Oxford Dictionary's definition of a *breakdown*, the thesis proposes the following definition for a *Scrum Process Breakdown:*

> *Any deviation from an ideal Scrum process (as per the Scrum methodology's guidelines) which leads to social challenges, conflict or disagreements in the GASD team.*

The study particularly focuses on sprint planning and retrospective meetings because of the rich collaboration, and the types of Scrum process breakdowns which are likely to occur between the participants during these activities. In particular, during *Sprint planning meetings,* stakeholders negotiate for the amount of requirements to be taken on board for a particular iteration (commonly known as a *sprint*). During a sprint planning meeting, business stakeholders would typically push for more requirements to be included in the sprint while software developers would negotiate to only include those requirements to which they can commit to successfully complete, based on team members' competencies and the complexity of the task at hand (Schuh, 2005).

During *Retrospective meetings*, team members report on issues experienced throughout the sprint, as well as what they believe went well and what they would like to continue doing. During these meetings, the team members also propose measures to improve the overall Scrum process and prevent these issues from occurring in the next sprint. This requires all team members' and stakeholders' participation, and they should all have the power to "voice-out" their opinion, without fear of retribution, and be listened to (Schuh, 2005).

Effective collaboration is said to be established under the following two circumstances:

(1) When the differences between the participants are leveraged to allow for the emergence of synergized and innovative solutions (Hardy, Lawrence, & Grant, 2005; Levina & Vaast, 2008).

(2) When the concerns of the various participants are addressed and balanced out (Hardy et al., 2005; Levina & Vaast, 2008).

However, effective collaboration is not easily achieved during GASD and social challenges are encountered while engaging in the above-mentioned work practices (e.g. Paasivaara et al., 2008; Ramesh, Cao, Mohan, & Xu, 2006; BjØrn & Ngwenyama, 2009). For example, during sprint planning meetings, some GASD team members have difficulty or are unable to engage in debates (Summers, 2008) and it is at times hard for them to establish good levels of interactions amongst each other (Danait, 2005; Smits & Pshigoda, 2007). GASD teams also have difficulty in being self-organised (as recommended by the agile principles (Koch, 2005)) and to negotiate (Batra, Xia, Van der Meer, & Dutta, 2010) during retrospective meetings. The social challenges reported are often attributed to differences in culture and lack of cultural understanding (Drummond & Unson, 2008; Summers, 2008; Paasivaara et al., 2008).

Collaboration within a GASD team can be investigated from either a human-centered (social) or a technical perspective. The social perspective requires the investigation of group dynamics whereas the technical perspective relates to a focus on the communication technologies required to allow for collaboration in this context. The presence of social challenges alludes to the fact that human-centred aspects of collaboration are exacerbated during GASD (Abbattista, Calefato, Gendarmi, & Lanubile, 2008). Indeed, it has been pointed out in Information Systems (IS) research that actions within an organisation are social in nature (Lyytinen & Ngwenyama, 1992; Ngwenyama & Lee, 1997; Ngwenyama, 1998) and coupled with the fact that agile methodologies integrate behavioural and social concerns into software development (Whitworth & Biddle, 2007), this points to the need for a better understanding of the social dynamics present during GASD.

1.3 Purpose of the Research

It has been established that various forms of social challenges occur during sprint planning and retrospective meetings within the GASD context (e.g. Summers, 2008;

Danait, 2005; Smits & Pshigoda, 2007; Young & Terashima, 2008; Batra et al., 2010). This study adopts the stance Scrum process breakdowns occur when the Scrum work practices followed by GASD teams deviate from the Scrum methodology's best practices (ideal Scrum). This leads to the occurrence of social challenges which induce conflict and disagreements between the team members.

For this study, the Scrum process breakdowns experienced during the sprint planning and the retrospective meetings will be identified. In addition, the Scrum process breakdowns experienced after these meetings, i.e. later on during the sprint when the decisions made during these meetings are overruled will also be investigated and reported. The study particularly focuses on spring planning and retrospective meetings as it has been reported in the literature that social challenges are often experienced when these two ceremonies are conducted during GASD (e.g. Drummond & Unson, 2008; Summers, 2008).

The study will also describe the social conditions under which these Scrum process breakdowns may occur during GASD. The particular stance adopted for investigating these Scrum process breakdowns are rooted in the notion that GASD team members collaborate in a context with overlapping fields of practice. In doing so, a better theoretical understanding of the social dynamics at play, giving rise to these Scrum process breakdowns within the GASD context, will be provided.

Scrum was considered relevant for this study because of its high popularity and usage rate. In particular, Scrum is one of the most popular agile software development methodology currently being employed in industry (Begel & Nagappan, 2007). For instance, in the *state of agile development survey* undertaken by VersionOne in 2012, Scrum was voted as the most employed methodology with 54% of the votes (VersionOne, 2012).

In this study, it is anticipated that a thorough understanding of the Scrum work practices followed by the GASD team members, particularly during the sprint planning and retrospective meetings should be obtained, in order to uncover the social conditions under which the Scrum process breakdowns occur. Schultze and Boland (2000) propose that in order to understand work practices, their circuit of reproduction should be investigated. They define the circuit of reproduction as the *"reciprocal, cyclical relationships through which practice creates and recreates the objectified structures and*

the conditions in which it occurs" (Schultze & Boland, 2000, p. 195). This study thus seeks to understand work practices relevant to the sprint planning meetings and retrospective meetings, by adopting this perspective. In particular, it will be demonstrated that by uncovering the social conditions under which practice creates and recreates the objectified structures, the social dynamics at play will be revealed. Bourdieu's Theory of Practice will be used as a lens to understand the above-mentioned GASD practices' circuit of reproduction. At the end of this study, a level 4 (Llewelyn, 2003), mid-range, descriptive theory (Gregor, 2006) will be proposed to describe the social conditions under which Scrum process breakdowns occur in GASD.

1.4 Research Objectives

The objectives of this study are:

a. To identify the forms of Scrum process breakdown which GASD teams may experience during and after sprint planning meetings
b. To identify the forms of Scrum process breakdown which GASD teams may experience during and after retrospective meetings
c. To describe the social conditions which may lead to Scrum process breakdowns during and after sprint planning meetings
d. To describe the social conditions which may lead to Scrum process breakdowns during and after retrospective meetings

1.5 Research Questions

The core research question is:

- What forms of Scrum process breakdown may GASD teams experience during and after sprint planning and retrospective meetings, and what social conditions may lead to these Scrum process breakdowns?

A number of secondary research questions have also been derived from the primary question, as specified below:

- What forms of Scrum process breakdown may GASD teams experience during and after sprint planning meetings?
- What forms of Scrum process breakdown may GASD teams experience during and after retrospective meetings?

- What social conditions may lead to Scrum process breakdowns during and after sprint planning meetings?
- What social conditions may lead to Scrum process breakdowns during and after retrospective meetings?

1.6 Importance of the Research

In an attempt to enhance GASD project success rate and team effectiveness in delivering high quality service, industry experts and researchers alike have to date focused most of their research endeavours on reporting lessons learnt on how to engage in GASD to achieve project and collaboration success, and on proposing mitigating strategies for the social challenges. Their aim has thus often been to propose sets of "best-practices" on how to engage in GASD work practices. As will be demonstrated in the literature review, the best-practice approach mostly acknowledges the existence of social challenges and is usually geared towards proposing solutions to these challenges, by reporting on the work practices which seemed to yield project success in GASD teams. Few studies have sought to understand the underlying social conditions giving rise to these challenges.

While the proposed "best-practices" are no doubt valuable to GASD, the social dynamics at play between the GASD team members while they engage in these work practices are yet to be understood. This study attempts to bridge this gap by providing a better understanding of the social conditions leading to breakdown in the Scrum process while engaging in GASD. In doing so, it is hoped that the assumptions made by the best practice approach will be highlighted, which could allow for more theoretically informed best-practices to be formulated in the future. An overview of how some of the current best practices could be adapted to avoid the breakdowns, by taking into consideration the social conditions under which they are implemented, is proposed in the conclusion.

The study is also important from a theoretical perspective. The current literature on GASD (both industry reports and academic papers) does provide insightful information on challenges experienced and best-practices required during GASD. However, as will be demonstrated in the literature review, the degree and quality of theorization requires improvement. Gregor (2006) posits that while formulating theory, it is important to be careful of the language being used, particularly while "differentiating between generality and predictive power, in defining the scope of a theory, and in the wording of the propositions" (Gregor, 2006, p. 634). The study seeks to improve on the degree of theorization currently being made in GASD literature and thus contributes to the GASD

research by formulating a level 4, mid-range, and descriptive theory based on recommendations from Gregor (2006) and Llewelyn (2003).

1.7 Research Motivation and Rationale Summary

This research study is motivated by the fact that the social dynamics present in GASD work practices have not been adequately investigated and theorised upon in current literature. The rationale behind the research motivation and purpose is summarised in Figure 1.1.

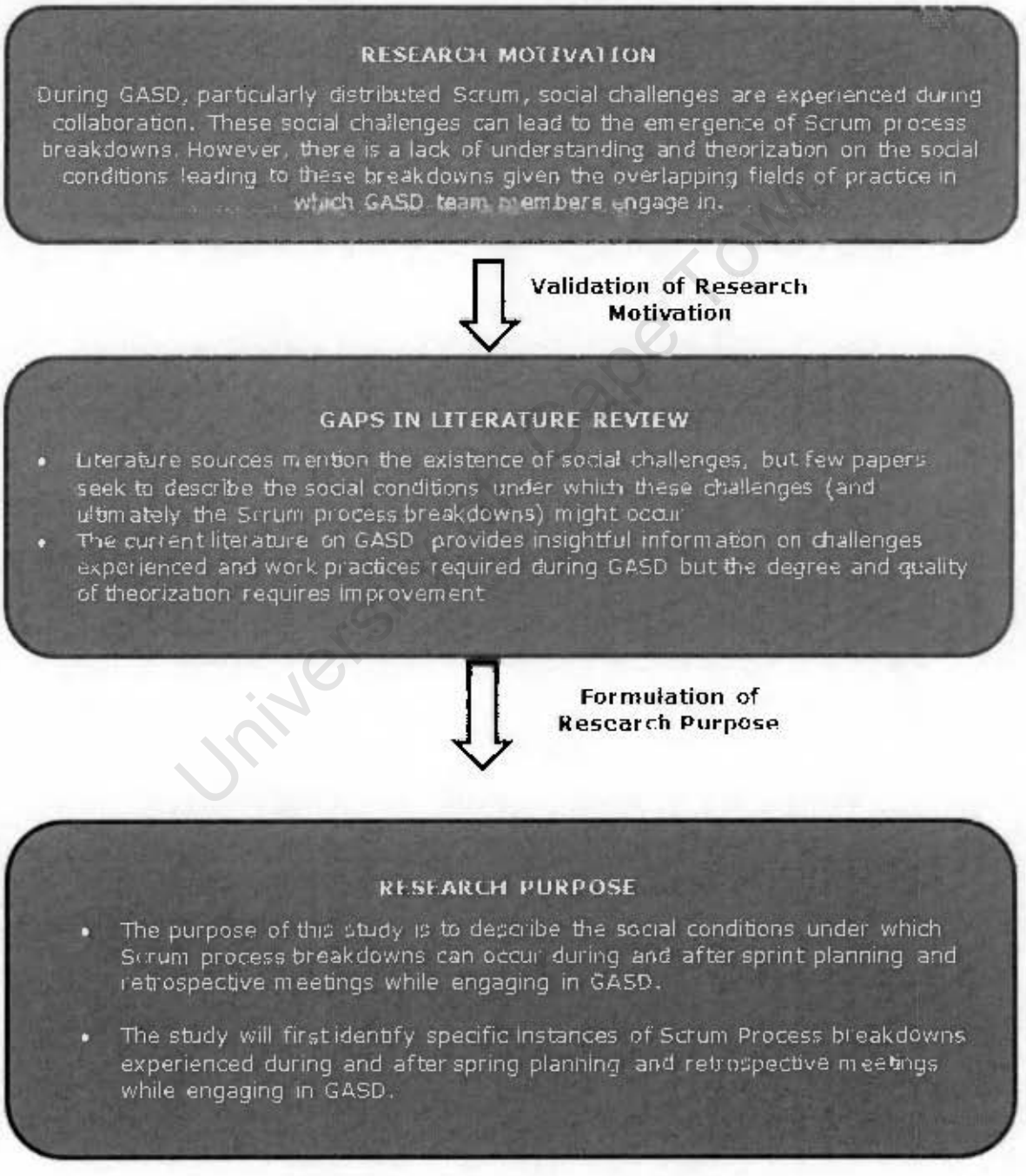


Figure 1-1 – Summary of Research Motivation and Rationale

1.8 Research Context

The research was undertaken in two GASD organisations employing Scrum as agile software development methodology to implement their software development projects. In both cases, the project was distributed across two sites in two different countries. This is further elaborated in the methodology and case description chapters.

1.9 Organisation of the Thesis

This thesis is organised into nine chapters. A comprehensive analysis of past literature on GASD is provided in Chapter Two. The Theory of Practice (Bourdieu, 1990), which form the theoretical basis for the study, is described in Chapter Three. Chapter Four then details the research methodology employed in the study. Chapters Five and Six detail the circuit of reproduction for the work practices identified in the first and second case studies respectively, while Chapter Seven is dedicated to the description of the identified Scrum process breakdowns and their proposed explanations. A theoretical discussion on the findings is further provided in Chapter Eight and the thesis is concluded in Chapter Nine.

2. LITERATURE REVIEW

The purpose of this study is to describe the social conditions under which Scrum process breakdowns occur during and after sprint planning and retrospective meetings, in GASD. This research objective is motivated by the claim that there is a lack of understanding and theorisation on the social dynamics induced by overlapping fields of practice characterising the GASD context.

The purpose of this literature review is to validate this research motivation. Firstly, it will be demonstrated that even though past studies report on particular forms of social challenges occurring in GASD, there is a lack of understanding of the social conditions giving rise to them. It will be demonstrated that instead, most research efforts in the field of GASD have been geared towards identifying "best-practices" to overcome these social challenges. It will also be demonstrated that even though the current literature on GASD provides insightful information on challenges experienced and best practices required during GASD, the degree and quality of theorization requires improvement.

A systematic review of the literature on GASD between 2006 and 2011 has been undertaken. In particular, the literature was surveyed to identify the social challenges experienced: (1) while following Agile principles in the GASD setting, (2) while engaging in Scrum work practices in the GASD setting, and (3) while following GASD work practices. This allowed for the identification of particular forms of social challenges from the literature, which emerge from Scrum process breakdowns. The proposed mitigating strategies to these social challenges have also been reported. In addition, the forms of theorisation proposed in each paper was evaluated using Gregor (2006)'s and Llewelyn (2003)'s theory classification schemes. A summary of these classification schemes has been provided in Appendices 9 and 10 respectively.

The literature review was conducted using Brereton, Kitchenham, Budgen, Turner, and Khalil (2007)'s guidelines on how to conduct a systematic literature review in the domain of software engineering. A manual search of conference proceedings, journal papers, and industry reports was conducted, resulting in the identification of forty-six papers from the following databases:

- ACM Digital Library
- Ebsco Host
- Emerald
- Google scholar
- IEEEXplore
- Compendex EI
- Wiley InterScience
- Elsevier Science Direct
- AIS eLibrary
- SpringerLink

The search terms employed included *Scrum, Agile, Distributed Scrum, Globally Distributed Scrum, Global Agile Software Development, Global Scrum, Challenges during Distributed Scrum, Distributed Scrum work practices, and Global Agile Practices.* In addition, whenever a relevant article was identified, the reference list was reviewed to identify secondary sources. A summary of the references for these forty-six papers is provided in Appendix 1.

2.1. Literature Review Structure

An overview of the agile principles followed by an in-depth description of Scrum work practices are presented in Sections 2.2 and 2.3 respectively. This serves not only to provide a background description of the research context, but also to identify from literature, what ideal sprint planning and retrospective meetings should be, both in terms of the meeting's structure and outcome. Any deviation from an ideal sprint planning and retrospective meeting yielding to the emergence of social challenges, conflict, and disagreements between the team members, points to a Scrum process breakdown.

Sections 2.4 to 2.6 describe the social challenges experienced while applying agile principles, Scrum work practices, and common GSD work practices in the GASD context, as well as the corresponding mitigating best-practices proposed in literature. The literature on GSD and GASD being similar in terms of the challenges and best-practices proposed, this literature review only focused on GASD. This also served to retain the focus on the Agile context. The gaps in the literature and the resulting research questions are presented in Sections 2.7 and 2.8 respectively.

2.2. The Agile Principles

Agile principles are not absolutely novel developments. Instead, they are derived from incremental and iterative software development paradigms which have been practised since the 1950s (Larman, 2004; Larman & Basili, 2003). The year 2001 witnessed the dawn of the "Agile Movement" in the software development industry (Beck, et al., 2001; Cockburn, 2002). As part of that "Agile Movement", an experienced group of software development practitioners formulated the Agile Manifesto (Agile Alliance, 2001), which described the agile principles forming the basis of agile methodologies.

Agile software development is based on practices which emphasise communication and soft skills as determinants of project success (Lindstrom & Jeffries, 2004). A development methodology is said to be agile when it is incremental (small software releases, with rapid cycles), cooperative (customer and developers working constantly together with close communication), straightforward (the methodology is easy to learn, to modify and is well documented), and adaptive (able to make last moment changes) (Ambler, 2002; Highsmith & Cockburn, 2001). Conboy (2009, p. 330) provided a different perspective on agility and defined it as "the readiness of a systems development methodology to rapidly or inherently create change, proactively or reactively embrace change, and learn from change while contributing to perceived customer value (economy, quality, and simplicity), through its components and relationships with its environment". The focal values (Agile Alliance, 2001) presented by the "agilists" are preferences for:

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

Agile software development is based on a set of 12 principles. Each principle relates to one of the agile values and represents the furthest that the supporters of various agile methodologies were able to agree upon (Cockburn, 2002; Ambler, 2005). The principles are (Koch, 2005, p. 5):

1. "Build projects around motivated individuals. Give them the environment and support needed, and trust them to get the job done"
2. "The best architectures, requirements, and designs emerge from self-organising teams"

3. "The most efficient and effective method of conveying information to and within a development team is face-to-face conversation"
4. "Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely"
5. "The highest priority during the software project development is to satisfy the customer through early and continuous delivery of valuable software"
6. "Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to shorter timescales"
7. "Working software is the primary measure of progress"
8. "Business people and developers should work together daily throughout the project"
9. "Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage"
10. "Continuous attention to technical excellence and good design enhances agility"
11. "Simplicity – the art of maximising the amount of work not done – is essential"
12. "At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behaviour accordingly"

2.3. Scrum

Scrum is an agile methodology which focuses on software project management (Larman, 2004). The methodology is useful in guiding software development under circumstances where different participants have conflicting interests and where requirements frequently change (Schuh, 2005). Scrum as a whole seeks to achieve energy, focus, clarity, and transparency during software development and within the team (Phalnikar, Deshpande, & Joshi, 2008; Sutherland, Viktorov, Blount, & Puntikov, 2007).

Scrum has been designed to leverage development speed, institute a performance driven culture, promote alignment of individual and organisational objectives, support shareholder value creation, and achieve consistent communication and performance (Sutherland et al., 2007). The constituents of the artefacts, roles, and work practices of Scrum are described in the following sub-sections.

2.3.1. Scrum Artefacts

During a sprint, team members self-organise, self-direct and might even change the functionality to be delivered within the sprint, so that the sprint goal is achieved (Schuh,

2005). In order to assist these team members in this task, Scrum proposes five main artefacts: the product backlog, the sprint backlog, the user story, the task board, and the burn-down chart (Cervone, 2010; Cocco, Mannaro, Concas G, & Marchesi, 2011).

The product backlog relates to an overall list of product requirements or features which still need to be completed for the project (Schuh, 2005). The purpose of the product backlog is to keep track of all work which remains to be done on the project as well as bug fixes. These product requirements are organised in order of priority. The product backlog can be considered as a roadmap for the development team (Schwaber & Beedle, 2002).

At the beginning of a sprint, the development team identifies a subset of the product backlog that will be completed during that iteration. The sprint backlog thus refers to that subset of the product backlog (Schuh, 2005). This subset of requirements is a more refined version of what is available from the product backlog (Cervone, 2010; Cocco et al, 2011). In addition, requirements are usually incorporated in the sprint backlog if they have been categorised as being of high priority in the product backlog (Schwaber & Beedle, 2002). The sprint backlog items are often represented as *User Stories* which are the predominant way for Scrum teams to express features. User stories are short, simple descriptions of the desired functionality told from the perspective of the user. For example: 'As a shopper, I can review the items in my shopping cart before checking out so that I can see what I've already selected' (Schwaber & Beedle, 2002). Once incorporated in the sprint backlog, user stories should not drastically change throughout the sprint to avoid negative impacts on the team's productivity rate and morale (Schwaber & Beedle, 2002). Only minor changes are allowed to the user stories provided that the sprint goal is still achieved.

The task board is a tool used by the Scrum development team to track all the work to be completed during a sprint and can be composed of several columns including: Task, To Do, In Process, To Verify, Done (Schwaber & Beedle, Agile Software Development with Scrum, 2002). Each task is moved from one column to the next throughout the sprint until it is fully completed and moved to the "Done" column (Schuh, 2005). Tasks are identified by breaking down specific features from the sprint backlog into more refined steps which could relate to analysis tasks, design tasks, coding or testing tasks, among others (Schwaber & Beedle, 2002). The task board is continuously updated throughout the sprint.

A burn-down chart is also maintained during the sprint. The burn-down chart is a graph indicating the amount of work left to be completed from the sprint backlog (Schuh, 2005). The graph is created by adding and plotting the remaining hours or days of work left from the sprint backlog estimates. A summary of the various Scrum artefacts is presented in Table 2.1.

Scrum Artefacts	Description
Product Backlog	An overall list of product requirements or features which still need to be completed for the project (Schuh, 2005)
Sprint Backlog	The sprint backlog refers to that subset of features from the product backlog which will be completed during the sprint (Schuh, 2005)
User Story	User Stories are short, simple descriptions of the desired functionality told from perspective of the user (Schuh, 2005)
Task Board	The task board is a tool used by the Scrum development team to track all the work to be completed during a sprint (Schwaber & Beedle, Agile Software Development with Scrum, 2002)
Burn-down Chart	The burn-down chart is a graph indicating the amount of work left to be completed from the sprint backlog (Schuh, 2005)

Table 2-1 - Summary of Scrum Artefacts

2.3.2. Roles within the Scrum Methodology

Schwaber and Beedle (2002) identified six roles in a Scrum team, namely: Scrum Master, Product Owner, development team, customers, management, and users. The roles are summarised in Table 2.2. The Scrum Master is the team leader and is responsible for liaising and facilitating communication between the customer, Product Owner, management and the development team (Schwaber & Beedle, 2002). The Scrum Master also ensures that Scrum work practices are followed and that values behind Scrum drive the process (Koch, 2005). One of the values behind Scrum is that team members are expected to be autonomous, implying that "people – processes" should be self-organising.

Scrum Role	Responsibilities
Scrum Master	Liaises and facilitates communication Ensures that Scrum practices are followed Ensures that values behind Scrum drive the process
Management	Allows development team to organise itself
Product Owner	Manages product backlog
Development Team	Responsible for delivering the system Commits to achieving a sprint
Customers	Exert pressure on the project only at the end of the sprint
Users	The users of the software application

Table 2-2 - Scrum Roles and Responsibilities (Source: Schwaber & Beedle, 2002)

Management is responsible for giving time and space to team members to organise themselves, as it is said that individuals are more inclined to take responsibility for tasks when they personally realise that particular tasks have to be completed (Schwaber & Beedle, 2002). The sense of responsibility is also further boosted when team members are given particular goals and objectives that they believe in.

The Product Owner is responsible for the product backlog (Schwaber & Beedle, 2002). Although all stakeholders are allowed to add tasks to the product backlog, only the Product Owner is allowed to prioritise the tasks.

The Scrum development team comprises members who are responsible for delivering the system (programmers, testers, analysts and technical writers). The development team commits to achieving a sprint (one iteration) goal, and "is accorded full authority to do whatever it decides is necessary to achieve the goal" (Schwaber & Beedle, 2002, p. 35). This level of autonomy is foreign to many organisations.

Customers can exert pressure on the project only at the end of the sprint. During the sprint, customers are not allowed to interfere and the development team is allowed enough time, clearance and trust to perform the work. Any issues reported by the customers at the end of a sprint are addressed in the next sprint (Schwaber & Beedle, 2002). At the beginning of a sprint, the objectives or purpose of that iteration are clearly defined. Customers are also allowed to regularly inspect the product while maintaining little variances from the objectives initially stated. Users are the individual human beings making use of the software application.

2.3.3. Scrum Work Practices

This section will describe the various work practices proposed by the Scrum methodology. In particular, an ideal sprint planning meeting, daily Scrum meeting, sprint review meeting and retrospective meeting will be described.

2.3.3.1. The Sprint Planning Meeting

At the beginning of a sprint, a sprint planning meeting is held where the Product Owner and the development team negotiate the features which will be tackled during that iteration. These features are selected from the product backlog and the development team should only select requirements which they can commit to complete during that sprint (Schuh, 2005). The sprint planning meeting can be divided into two individual meetings namely *sprint planning meeting 1* and *sprint planning meeting 2.*

In the sprint planning meeting 1, the sprint schedule (start date and end date) is defined and the product backlog is reviewed and reprioritised to negotiate on what features will be developed during the sprint, and compose the sprint backlog (Schuh, 2005) The sprint goal is also agreed during that meeting, and relate to a summary of the purpose of the sprint (Beedle, Coplien, Sutherland, Østergaard, Aguiar, & Schwaber, 2010). The negotiation process pertaining to user story selection is based on the current content of the product backlog, information about the latest product increment obtained from the customer, the team capacity and performance (Schwaber & Beedle, 2002). It is important for the development team and the Product Owner to engage in the following activities during sprint planning meeting 1 (Schwaber & Beedle, 2002):

- The Product Owner identifies items of high priority from the product backlog which he/she would like to have completed during the upcoming sprint
- The development team and the Product Owner go through each item methodically to ensure that they understand the product requirement.
- In particular the Product Owner explains the feature from a functional perspective
- The development team acquires as much information about each task by asking questions in order to establish how the feature should work.

Estimation is an important activity which the development team needs to engage in while determining which tasks to include in the sprint. During the estimation process, the development team assesses the size of a user story. This can involve determining the amount of effort required to complete that story (complexity), or the costs involved

(Schwaber & Beedle, 2002). This is often achieved during estimation games. Scrum specifies that the entire development team should engage in the estimation process and each team member should be given the chance to express their opinion on the tasks' complexity. Scrum does not prescribe a specific way for teams to classify the stories while undertaking the estimation process but common schemes include (1) numeric sizing (2) T-shirts sizes (XS, S, M, L, XL, XXL, XXXL) (3) the Fibonacci sequence (Schwaber & Beedle, 2002).

Development teams can also play games like the *Planning Poker Game* to encourage participation from all team members during the estimation process and make the process faster and more accurate. The outcome of this process should be a common agreement on the features to be included in the sprint backlog and a thorough and accurate understanding of what the feature entails (Schwaber & Beedle, 2002). This agreement should be realistic and should satisfy everyone. It is expected that the requirements' content and priorities should not change drastically during the sprint. This implies that no changes in requirements, which go against the sprint goal, should be requested by the customers as any major change could be detrimental to the team performance and morale (Schwaber & Beedle, 2002).

Once that the development team and the Product Owner have agreed on the sprint backlog items, the sprint planning meeting 2 is held to determine how each story will be completed. In particular, during this meeting, each user story is broken down into individual tasks, making sure that all work required to complete that story is accounted for (Schwaber & Beedle, 2002). For example, individual tasks could relate to analysis tasks, design tasks, coding, testing, code reviews etc. The development team also estimates the duration of each task, either in days or in hours. These individual tasks are also included in the task board and tracked throughout the sprint (Schwaber & Beedle, 2002).

2.3.3.2. The Daily Scrum Meeting

In Scrum, communication is facilitated through daily Scrum meetings held between the Scrum Master, the Product Owner and the development team for them to synchronise. This ensures that all parties are aware of the project progress and any roadblocks/impediments which might arise. The daily Scrum meetings last for about fifteen minutes during which team members are expected to answer three questions as

shown below (Schuh, 2005) and any impediments should be resolved within 24 hours of being raised or notice.

(1) What have you done yesterday?

(2) What will you do today?

(3) What got in your way of doing your work?

2.3.3.3. The Sprint Review Meeting

Each sprint ends with a potentially shippable product increment (Schuh, 2005) and is followed by a sprint review meeting. The meeting provides a picture of the progress made and lays the foundation for the next sprint planning meeting (Schuh, 2005). The aim of the sprint review meeting is to allow the Scrum team to discuss about what has been achieved during the previous sprint and to obtain feedback on the product or prototype. The feedback obtained from that meeting allows the development team and the Product Owner to update the product backlog and decide on the necessary requirements to be completed for the next sprint. The meeting is informal and can take the form of a demo where the development team showcases the work done and answers questions. Either prototypes and/or potential shippable product increments could be demoed during these meetings (Schwaber & Beedle, 2002).

2.3.3.4. The Retrospective Meeting

A separate sprint retrospective is also held after the sprint review meeting. During this meeting, which can last for three hours, the Scrum Master encourages the development team to reflect on their current work practices and to revise them accordingly. The aim is to make the Scrum process more enjoyable and effective in the next sprint (Schwaber & Beedle, 2002).

In particular, during that reflection the development team should identify and prioritise what went well, what they could have done better and should be doing differently, and what they should continue doing (Schwaber & Beedle, 2002). The development team is expected to focus on the people, relationships, process and tools. For instance, they could inspect their actual Scrum team composition, their meeting arrangements, tools, methods of communication and practices employed to turn a sprint backlog item into a shippable increment. Upon completing a retrospective meeting, the development team should have identified a list of actionable sets of measures to be followed to improve the Scrum process for the next sprint (Schwaber & Beedle, 2002). The Scrum process is illustrated in Figure 2.1.

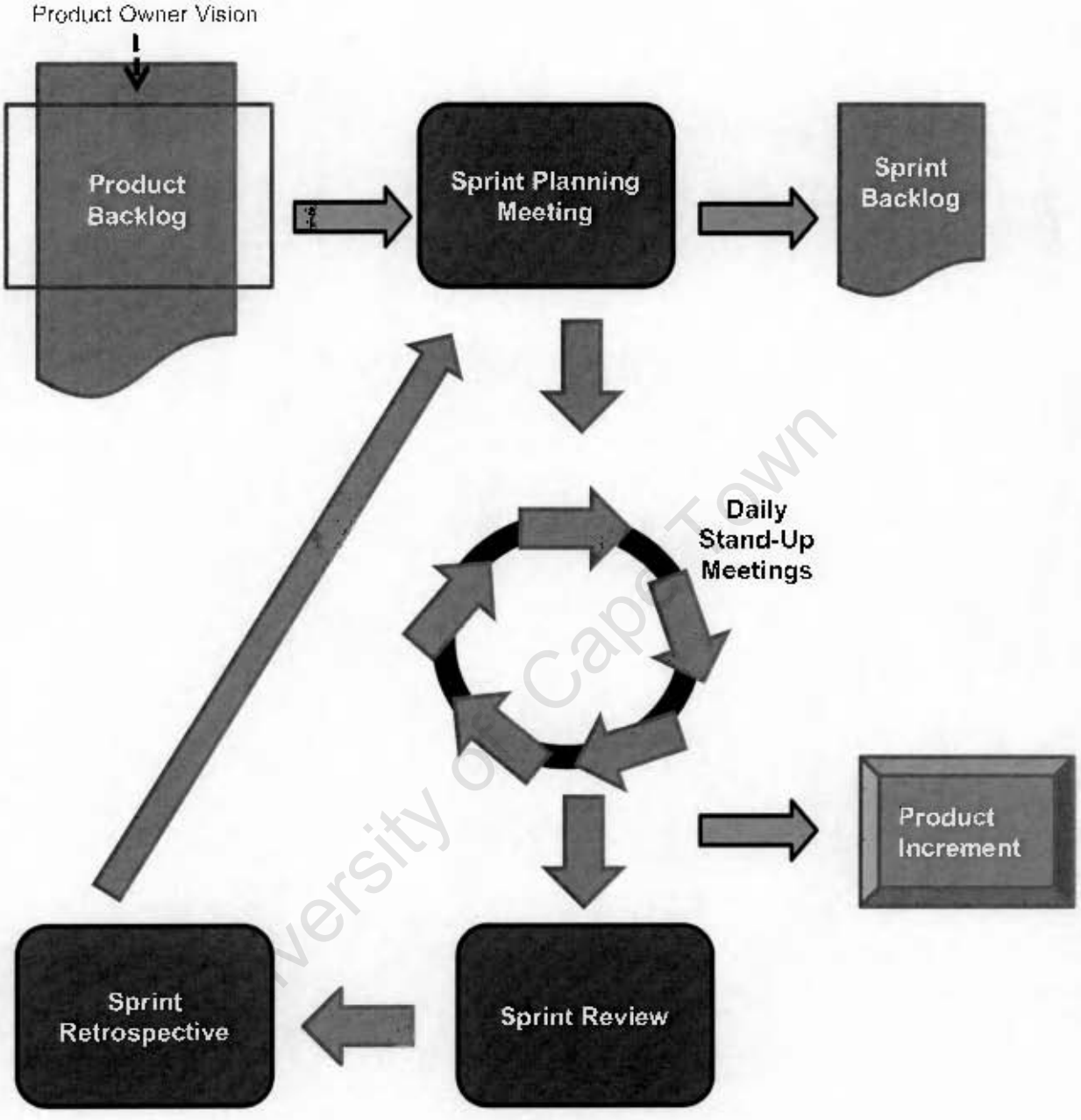


Figure 2-1 - The Scrum Process

2.4. Social Challenges Experienced while Following Agile Principles During GASD

The 12 agile principles which form the foundation of agile methodologies were described in Section 2.2. Upon reviewing the literature, it was found that seven of these twelve principles could be impacted by various forms of social challenges in the GASD context. These are further described in the following sub-sections.

2.4.1. Motivated Team Members (Principle #1)

The first agile principle states that projects should be built around motivated individuals and that these individuals should be given the support to get the job done (Koch, 2005). Past studies have found that when GASD is implemented in an onshore/offshore setup, motivation cannot be easily controlled and enhanced at the offshore site(s) (Batra, 2009; Batra, Xia, Van der Meer, & Dutta, 2010). Often, the lack of motivation at the offshore site was attributed to geographical distance between the team members (Batra, 2009; Batra, Xia, Van der Meer, & Dutta, 2010). However Phalnikar et al., (2008) found that, it was a lack of interpersonal relationships, more than geographical distance, that accounted for low motivation.

The use of short iterations, frequent builds, and continuous integration was proposed as a solution to the lack of motivation issue (Dullemond, van Gamaren, & van Solingen, 2009). This argument was built on the premise that through the use of short iterations and frequent builds, team members would be able to visualise progress and obtain feedback on their work, which would contribute to motivating them (Dullemond et al., 2009).

2.4.2. Self-Organising Teams (Principle #2)

The second principle states that "the best architectures, requirements, and designs emerge from self-organising teams" (Koch, 2005, p. 5). Self-organisation implies that there is no appointed leader or manager in the team. As mentioned by Hundermark (2009, p. 6), self-organised teams "are given full autonomy and carry ... greater responsibility for delivery according to their own commitments.... They are encouraged to take reasonable risks and to learn through failure and self-reflection". Past studies have shown that self-organised teams might not always be possible due to cultural disparities between project stakeholders (Batra, 2009), hence the emergence of social challenges. The development team was at times also found to misunderstand the concept of "self-organising team" in that context, leading to an "abuse [of] freedom" (Batra, et al.,

2010). Coupled with the fact that the relationship between the development team and the customers often remained rather formal (Batra et al., 2010), self-organisation was reported to be harder to establish within the GASD context.

Various practices have been put forward as a solution to that issue. For instance, it was suggested that self-organisation could be enhanced if meetings were recorded and replayed later during the sprint (Danait, 2005). The support of management towards the team's effort to self-organise, along with some degree of guidance, was also found to be important (Berczuk, 2007). The guidance could be provided through the use of a facilitator who would assist and empower the team to solve their own problems (Therrien, 2008).

2.4.3. Face-to-Face Conversations (Principle #3)

According to the third principle, face-to-face conversation is the most efficient and effective method of conveying information to and within a development team (Koch, 2005). However, because of geographical and temporal distance, this form of communication was found to be difficult in GASD. As open communication is crucial in a distributed agile project and since the distributed team might not physically meet regularly, it was suggested that project-related information could be passed on via communication technology (Phalnikar et al., 2008). Team members would thus make use of communication and collaboration tools to simulate the open-communication process (Batra, 2009; Hossain, et al., 2011; Ramesh, et al., 2006). The lack of face-to-face communication was also said to be improved through synchronised work hours, balanced coordination, constant communication and the use of informal communication through formal channels (Ramesh et al., 2006).

Thanks to technological advancements, there are now a wide range of options to effectively mimic face-to-face experience while operating in a distributed setup. A summary of these tools and their purpose is provided in Table 2.3.

Communication Tool	Purpose
Conference Phones Web Cams	Made available in tea rooms for impromptu meetings (Ramesh, et al., 2006)
Video Conference Applications Web Conferencing applications + projector	Achieve real time interactive collaboration (Danait, 2005)
Virtual White-board	Used to understand and chalk out design, code repository walk-through, pair programming and kick off meetings (Danait, 2005)

Table 2-3 - Purpose of Communication Tools

However, the use of these tools was found to only facilitate business communication and not organisational communication (Batra, 2009). Business communication includes formal and planned messages as represented in emails, memos, reports, Web sites etc. Organisational communication relates to business communication but also informal day-to-day interactions between stakeholders (Batra, 2009). The use of documentation (Ramesh et al., 2006) and blogs (Abbalisto et al., 2008) was proposed to solve the informal communication challenges. The introduction of a new team member to a dispersed team was also found to be problematic without face-to-face interaction and Danait (2005) proposed the use of web conference tools to facilitate this process (Danait, 2005).

As means to reduce the negative impact of distance during the requirements gathering stage of the project, it was recommended that the person representing the customer should be defined up front. This person should be able to make decisions pertaining to project functionality and scope, should be readily available and should have a strong interest in the project (Layman, Williams, Damian, & Bures, 2006).

According to Shrivastava and Date (2010) the agile precept of having continuous code integration is an important mechanism to enhance the degree and effectiveness of communication between the distributed team members. Communication is enhanced because iterative software development and continuous code integration allow all team members to develop a common short term goal.

2.4.4. Working Software (Principle #7)

According to past studies, the seventh agile principle which states that "working software is the primary measure of progress" (Koch, 2005), also appeared to be impacted by

social challenges in GASD. For instance, Danait (2005) reported that working software and stable build at the end of every work day can be established and maintained through collective ownership of code. If collective code ownership is established, team members would not "point fingers" during code break and would instead collectively focus on maintaining a stable build. However, collective ownership is harder to establish in a GASD setting because of possible cultural values which might clash with this principle (Dullemond et al., 2009). For instance Berteig (2007) reported that the idea of collective code ownership was perceived by participants of an agile development course as being similar to communism.

2.4.5. Close Collaboration between Developers and Customers (Principle #8)

Past studies reported that social challenges were also experienced while implementing the eighth agile principle, which states that "business people and developers should work together daily throughout the project" (Koch, 2005, p.5). In GASD, the need to closely work together is especially important during the requirements definition stage and while negotiating acceptable levels of quality (Ramesh et al., 2006). This agile principle relies heavily on social interactions and is not easily implementable in the GASD setting. In particular, because of limited ability to control the activities of the remote teams, GASD often tends to rely on fixed, upfront commitments on quality requirements as opposed to being able to negotiate (Ramesh et al., 2006). For example, Andrzeevski (2007) reported that they experienced issues pertaining to collaboration with the customer in the GASD setting. These issues occurred while: (1) undertaking planning game with the remote customer, and (2) providing daily feedback. To resolve these challenges, it was suggested that customer visits should be organised during the planning games to produce clearer requirements, more optimistic estimations and less rework during the iterations (Andrzeevski, 2007).

2.4.6. Welcoming Changes in Requirements (Principle #9)

Past studies have reported social challenges while implementing the ninth principle in the GASD setting. This principle states that changes in the requirements should be welcomed, even late in the development process (Koch, 2005). For instance, in one specific project investigated by Batra et al., (2010), it was found that contract could not be easily negotiated to accommodate for late changes to requirements. It was also found that because of geographical distance, the high communication rate required to manage the requirements change process properly was not easily achievable (Paasivaara & Lassenius, 2006).

2.4.7. Reflection to Improve Team Effectiveness (Principle #12)

The twelfth agile principle states that agile team members should regularly reflect on how to become more effective and should adjust their behaviour accordingly (Koch, 2005). Past studies have identified some forms of social challenges while applying this principle in the GASD context. For instance, Batra (2009) stated that this practice is not always feasible because of organisational culture. It was also stated that because of geographical distance and some particular team structures, there could be an imbalance of power between the distributed agile team members which might lead to feelings of frustration. The power might be skewed towards the site where the Product Owner and the Scrum Master reside.

Therrien (2008) proposed that in order to overcome this issue, offshore team members should be empowered and should be encouraged to fully participate in the process (Therrien, 2008). It was also proposed that decision making should be decentralised. In particular, part of the decision making should be performed by the developers while management is expected to remove impediments hindering developers from getting the job done (Dullemond et al., 2009). However, it has been acknowledged that decentralised decision making is hard to establish in the GASD setting because people can have different attitudes towards hierarchy and also because the developer sitting at a remote site might not have all the information required to make an informed decision (Dullemond et al., 2009). Appendix 3 summarises the social challenges experienced while applying Agile principles in the GASD context as well as the corresponding mitigating strategy, as identified from literature.

2.5. Social Challenges Experienced while Using Scrum Work Practices in GASD

As with all agile methodologies, Scrum relies on strong social ties which are not easily produced and maintained in the GASD setting (Sutherland et al., 2007). The business environment is changing rapidly and, coupled with distribution, the use of Scrum becomes problematic (Berczuk, 2007). According to current literature, the Scrum work practices in which social challenges were commonly experienced in GASD are *sprint planning meetings, daily stand-up meetings, retrospective meetings,* and *sprint reviews.*

2.5.1. Sprint Planning Meetings

Past studies have reported that social challenges are experienced during the sprint planning meetings in GASD. For instance, cultural and language differences at times

caused GASD participants to remain silent and not fully participate in the sprint planning meetings. It was also reported that GASD participants found it difficult to recognise speakers during conference calls when not seeing their faces (Paasivaara, Durasiewicz, & Lassenius, 2008). Studies have also shown that some GASD team members found it difficult to engage in debates and were not able to challenge customers who requested a large number of requirements to be completed during the sprint.

These social challenges are often attributed to culture (Summers, 2008) and a lack of understanding of cultural norms across the sites, and are seen to introduce feelings of frustration among the GASD team members. For instance, Drummond and Unson (2008) reported on a situation where offshore team members were requested to attend late sprint planning meetings in timeslots during which they would normally be spending time with their families. This was seen to lead to strong feelings of frustration in the team. Hossain, Bannerman and Jeffery (2011) reported that in order to avoid such feelings of frustrations, various holidays at the diverse sites should be taken into account during the sprint planning meetings.

Studies have also found that sprint planning or kick off meetings could be hard to undertake in the GASD setting as the systems analyst might not be present with the development team as it is not a defined Scrum role (Danait, 2005). It can also become more difficult to facilitate the interaction between the Product Owner and the distributed team under these conditions. For example, the situation was seen to become more complex if for a large project, there are several Product Owners at different sites, each responsible for different project components (Smits & Pshigoda, 2007).

Danait (2005) further stated that this type of social challenge occurs because of "geographical distance" between the Product Owner and the development team. The use of web-conferencing tools during kick-off meetings was thus proposed to bridge the physical distance between these project participants (Danait, 2005). In Danait (2005)'s study, the meetings were recorded and replayed as a means of eliminating the need for the systems analyst to be constantly physically present.

It has also been proposed that participants from the dispersed sites should be brought together during early project stages (e.g. release planning meetings). For example, Smits and Pshigoda (2007) report that in their project, the GASD participants used this

time spent together to bond and create a team spirit and work through questions and issues.

Estimation is an important aspect of the sprint planning meeting and studies have found that GASD team members might not have enough experience with conducting estimation games across shores (Summers, 2008). Under such circumstances, Summers (2008) propose that a facilitator could be present at both sites to help with the process (Summers, 2008). Summers (2008) found that as the team became more experienced, the facilitator was no longer needed (Summers, 2008).

2.5.2. Daily Stand-Up Meetings

Past studies have found that social challenges are experienced when GASD participants engage in daily stand-up meetings. For instance, because of cultural differences, some participants may find it difficult to report on impediments that they experience (Paasivaara et al., 2008). There might also be information overload during these meetings (Young & Terashima, 2008). In order to reduce information overload Young and Terashima (2008) suggest that, at the beginning of each meeting, there should be a small "greeting period", during which each GASD team member would discuss ordinary topics like the weather, the people's wellness, and by showing the environment in which they are working via the webcam (Paasivaara et al., 2008; Sutherland et al., 2007).

Summers (2008) also suggested the use of user stories to increase communication. For example, when the communication was found to be ineffective in the GASD context, the use of user stories was seen to encourage dispersed team members to seek further clarifications. This further leveraged communication and contributed to a correct understanding of the requirements. These conversations also further helped improve the relationship between the offshore team members and the business (Summers, 2008).

2.5.3. Retrospective and Sprint Review Meetings

The retrospective meeting also appears to be harder to implement during GASD because of social challenges. Drummond and Unson (2008) reported a case where retrospective meetings did not successfully reveal team frustration which had occurred because of socio-cultural distance. Instead, these feelings of frustration were revealed through one-on-one informal conversation with the Scrum coach. Summers (2008) reported a similar situation whereby even though team members reported on what went well and what

went wrong during the retrospective meetings, they did not however engage in activities to drive that process forward.

On the other hand, Shrivastava and Date (2010) reported that sprint reviews could have a positive impact on the effectiveness of communication between the GASD team members. They posit that, through the use of sprint reviews, communication could be improved as GASD team members would be enticed to openly discuss the features and requirements dependencies for the project (Shrivastava & Date, 2010). In this case, the practice of having sprint reviews was perceived as having a positive impact on communication effectiveness, whereby whenever sprint reviews were held communication was enhanced. Appendix 2 summarises the social challenges experienced while engaging in Scrum work practices in the GASD context as identified from literature.

2.6. Social Challenges Experienced while Following Common GSD Work Practices

The following subsections describe the social challenges encountered while following widely used GASD work practices as reported from past literature.

2.6.1. Communication Practices

Communication is important to software development as a whole (Curtis, Krasner & Iscoe, 1988) but because of geographical, temporal, and socio-cultural distance, it has often been described as a crucial aspect of GASD in need of careful management (Phalnikar et al., 2009). As a consequence of socio-cultural distance, it was reported that GASD participants might have varying attitudes toward hierarchy, sense of time, communication style, need for structure (Holmström, Fitzgerald, Ågerfalk, & Ó. Conchúir, 2006; Sutherland, Schoonheim, Rustenburg, & Rijk, 2008), and work styles (Sutherland et al., 2007). Hence, there is a risk that GASD participants might misunderstand each other, thus leading to communication breakdown (Holmström et al., 2006). Furthermore, the communication breakdown and the delays in obtaining responses can cause frustration among team members (Holmström, et al., 2006).

BjØrn and Ngwenyama (2009) attribute this communication breakdown to a lack of shared meaning between the participants. Shared meaning helps team members make sense of each other's actions and is built over time and via face-to-face interactions. Shared meaning is more difficult to create in the GASD context as participants mostly communicate through communication technology. BjØrn and Ngwenyama (2009) posit

that there can be three levels of shared meaning at which communication breakdown can occur: lifeworld, organisation, and work practice.

Various GASD practices were proposed to mitigate the communication breakdown. For instance, Layman et al. (2006) proposed that key team members from one site should be physically located with the other team members at the other dispersed site(s) to facilitate and enhance the communication conduit. Layman et al., (2006) also recommend prompt, useful and conclusive responses to emails in situations where face-to-face and synchronous forms of communication are not feasible. BjØrn and Ngwenyama (2009) however challenge this statement by affirming that the use of emails is not enough in that context.

It has also been proposed that to prevent misunderstanding and language issues, there should only be one point of contact between the team members and the customer (Andrzeevski, 2007). That key person could be 'the voice of the team' (Therrien, 2008). Sutherland et al., (2008) also pointed out that the Scrum Masters should promote a culture of openness and direct communication in the team.

Other mechanisms proposed to increase communication effectiveness in the GASD setting include reducing time differences through the use of flexitime, addressing language barriers, developing trusted relationships, increasing effective formal communication, and increasing effective informal communication (Dorairaj, Noble, & Malik, 2011). Vax and Michaud (2008) also propose that in the event of the use of flexitime, daily scrum meetings can be reduced to only three times weekly in order to reduce the strain of late nights/early mornings every day. Longer overlaps can only happen on days where flexitime occurs. Communication tools can also be employed to minimise distance and maximise visibility on who is available in the office and who is not. The use of technology also allows for project visibility whereby information can be available to all about tasks, individuals, and the team, across the sites (Dullemond et al., 2009).

2.6.2. Coordination Practices

The process of coordination consists of splitting tasks into subtasks, allocating tasks to team members, ordering of the subtasks over time, and combining or integrating the results of the subtasks (Mockus & Herbsleb, 2001). Various forms of social challenges during coordination have been reported in past literature. For instance, because of

geographical distance, it was noted that GASD team members might not have enough informal contact, which could lead to a lack of task awareness, which in turn hinders coordination (Holmström et al., 2006).

Also, client nuances and priorities are not easily communicated to dispersed team members (Sutherland et al., 2008) and it is harder to convey a common vision and strategy across the sites (Holmström et al., 2006). There can also be a lack of shared context and knowledge across the dispersed sites (Simons, 2002). These might result in forms of implicit assumptions on the part of the development team which might not be identified as fast as in a collocated environment (Paasivaara & Lassenius, 2006). Miscommunication and lack of clarity can also occur while sending reports and updates among the team members (Robarts, 2008) and cultural differences might lead to inconsistent work practices across the dispersed sites which could in turn impinge on effective coordination (Holmström et al., 2006).

In order to mitigate the negative impact of these challenges, it has been proposed that work can be divided into modules which can be developed independently at the various sites to minimise communication between stakeholders (Paasivaara & Lassenius, 2006). These modules can then be integrated incrementally (Paasivaara & Lassenius, 2006). Alternatively, in order to create a common understanding of the work context and task priorities, regular travel, always-on Skype connections, and the use of project news gazette can be promoted (Sutherland et al., 2008). The use of unit tests has also been proposed to communicate the status of the project to the entire team (Berczuk, 2007).

Robarts (2008) proposed a different perspective on how to mitigate coordination issues, which takes into consideration an important social element, i.e. motivation. In particular, Robarts (2008) suggests that to introduce more clarity while writing reports for the dispersed team members, it is important to understand the recipient's motivation for reading the report.

2.6.3. Team Cohesion Practices

In a study undertaken by Whitworth and Biddle (2007) on the socio-psychological experience of agile teams, it was reported that team cohesion is an important requirement for agile environments. Various social challenges have been identified while seeking to achieve and maintain team cohesion in GASD teams. For example, it was reported that it is hard to establish a feeling of trust and "teamness" among the

dispersed team members even though "cross-site relationship" might be established (Holmström et al., 2006; Ramesh et al., 2006). In particular, remote team members find it hard to trust or have faith in the good intentions of their remote colleagues (Dullemond et al., 2008). GASD participants might also feel that they could lose their jobs or be relocated to other sites (Holmström et al., 2006). This situation, attributed to geographical distance and the time spent without seeing the dispersed team members for many months, further reduces motivation, team velocity, and work quality (Andrzeevski, 2007).

Because of cultural differences, GSD participants might also have different perceptions on issues around authority and hierarchy which could in turn undermine team morale (Holmström et al., 2006). In particular, the introduction of agile methodologies might disturb the current pattern of command and control at the various sites (Paasivaara & Lassenius, 2006). Some sites might have more hierarchical management structures in comparison to others with flatter structures, and this could further introduce managerial and social issues (Sutherland et al., 2008). Distributed team members can also have limited understanding of the importance of teamwork (Dorairaj et al., 2011) and when a feeling of "camaraderie" is established among the developers, it might be due to the high degree of collectivism in that culture (Sison & Yang, 2007).

Various practices have been put forward to promote trust among the GASD team members. For instance, it was proposed that in order to establish a sense of shared purpose and common objectives throughout the dispersed teams, frequent visits of distributed partners and sponsor should be organised and efforts should be made to build a cohesive team culture (Cottmeyer, 2008; Ramesh et al., 2006; Sutherland et al., 2008). It was also said that during these visits each partner should spend time listening to each other, communicating their goals and business model, as well as getting to know each other's leadership style (Cottmeyer, 2008). Visits have been reported to be particularly important during project kick-offs (Dorairaj et al., 2010) and for quarterly release planning meetings (Drummond & Unson, 2008).

Even when not meeting face-to-face, each individual should be reassured that all team members share the same team goal and are aware of the team activity. Feedback and awareness should be elicited from the whole team as opposed to individually or through partial team communication (Whitworth & Biddle, 2007). In addition, Urdangarin,

Fernandes, Avritzer, and Paulish (2008) report that eXtreme Programming (XP) can also be used to enhance trust.

A less expensive option than site visits which has been proposed is the use of social network profiles to develop digital identities, establish connections and build trust. This can even help in establishing a common culture among the project stakeholders (Abbatista et al., 2008). Dorairaj et al., (2010) propose that the use of video conferencing is even more useful to improve trust than email or wikis as it conveys information about body language. However, it has been reported that that even though technologically mediated forms of communication are useful in the GASD context, it hinders the team maturity process (Drummond & Unson, 2008). In addition, BjØrn and Ngwenyama (2009) state that face-to-face communication can also induce communication breakdown, since it is during co-located events that major discontinuities at the lifeworld level become salient.

Trust was also fostered in GASD when team members respected and supported each other. For example, it was shown that when some team members were willing to adjust their working hours to maintain an open communication line across the sites, trust was established (Vax & Michaud, 2008). Treating offshore team members in the same manner as full team members was also found to promote trust in the dispersed team (Cottmeyer, 2008).

Past studies have also shown that team members should also work towards maintaining the trust that has already been established. For instance, Cottemeyer (2008) reported on a particular case where onshore team members were aware of the "harsh" measures which would be taken towards offshore team members if issues were escalated to their upper management, and thus did not escalate these issues unless absolutely necessary. They rather focused on maintaining the trust and establishing a safe working environment for their offshore team members.

Another important aspect of team management which needs to be taken into consideration in the GASD setting is the introduction of new team member, as this situation could create uncertainties for the existing team members. These uncertainties relate to how the new person will catch up with the rest of the group or how he or she will contribute to the project (Young & Terashima, 2008). Young and Terashima (2008) report that under these circumstances, an introductory video conference session can be

held to introduce the person to the coding standards, programming practices, documentation, and testing expectations. The new team member can also describe his or her personal skills to the rest of the team. According to Uy and Ioannou (2008), team cohesion and performance could be achieved by tackling the following social challenges: absence of trust, fear of conflict, lack of commitment, avoidance of accountability, and inattention to results.

2.6.4. Knowledge Sharing Practices

In the GASD setting, knowledge sharing of the team members' experiences, skills, and methods used, is not easily achieved because of various forms of distance. And yet, knowledge sharing is an important procedure to improve development process effectiveness, in particular to improve cost reduction and the execution of redundant work (Shrivastava & Date, 2010). Studies to date do not appear to report on specific social challenges that hinder knowledge sharing. Instead they propose several mitigating strategies to enhance this process.

For instance, it has been mentioned that knowledge sharing can be facilitated through the use of a product/process repository to store contents from various sources like emails and online discussion threads (Shrivastava & Date, 2010). Carefully planned rotation of team members across the sites could also allow them to meet and learn from senior team members (Cottmeyer, 2008). Appendix 4 summarises the social challenges experienced while employing GASD work practices and the proposed mitigating practices.

2.7. Gaps in the Literature

Appendix 5 provides a visual representation of the various angles through which GASD has been studied from 2006 to 2011. The figure shows that the papers reviewed focused on two main aspects of GASD:

(1) Identifying social challenges experienced while engaging in GASD.
(2) Identifying the social challenges, the papers proposed mitigating strategies to reduce their possible negative impact.

It appears that few studies sought to identify how breakdowns in the social interactions between the GASD participants lead to these social challenges and Scrum process breakdowns. These social challenges appeared to be reported (1) directly observable phenomena and as (2) consequences of geographical, cultural, and temporal distances.

Thus, there is a need for a deeper understanding of the social conditions giving rise to these challenges and ultimately, Scrum process breakdowns.

Most of the papers reporting evidence of social challenges and corresponding mitigating strategies were industry reports which were written as application descriptions, in line with Benbasat, Goldstein, and Mead (1987). (See Appendix 8) Application descriptions are accounts written by practitioners to describe their experiences while implementing an application. In doing so, most of the time they focus on successful projects and conclude by proposing a list of "Do's" and "Don'ts" to be used by other practitioners wishing to implement a similar application (Benbasat, Goldstein, & Mead, 1987). These industry reports often provided interesting insights, thus calling for scientific investigations to validate these claims.

The review of the relevant papers also assessed their level of theorisation based on the guidelines from Gregor (2006) and Llewelyn (2003). Gregor (2006) proposes that theories can be classified into five types: Analysis, Explanation, Prediction, Explanation and Prediction, Design and Action. Llewelyn (2003) proposes five forms of theorizing: Metaphor Theories (Level 1), Differentiation Theories (Level 2), Concepts Theories (Level 3), Theorizing Settings (Level 4), and Theorizing Structures (Level 5). Each classification scheme has been summarized in Appendices 6 and 7.

Gaps pertaining to the degree of theorisation were also identified from the literature review. Firstly, a large proportion of the papers reviewed were at the analysis level (15 out of 46 as shown in Appendix Nine) according to the Gregor (2006) categorisation schemes and some could not even be classified into any of Llewelyn's categories because of their poor level of abstraction and generalizability (12 out of 46 as shown in Appendix Ten). Those formulated at the explanation (14 out of 46), prediction (3 out of 46), explanation and prediction (4 out of 46), and design and action (16 out of 46) levels (Gregor, 2006) or at Levels one (1 out of 46), Two (2 out of 46), Three (10 out of 46), and four (21 out of 46) (Llewelyn, 2003) were not always correctly phrased. In particular, even though it was possible to get a general sense of what the theory was about, constructs were not clearly defined in the statements of relationships. When represented graphically, the relationship between the constructs was not described in words. In addition, the scope was almost never mentioned, which makes it difficult to define the theories' relevance boundary. Given the gaps identified in the literature, the research questions described in the following sub-section were formulated.

2.8. Research Questions

The core research question is:

- What forms of Scrum process breakdown may GASD teams experience during and after sprint planning and retrospective meetings, and what social conditions may lead to these Scrum process breakdowns?

A number of secondary research questions have also been derived, as specified below.

- What forms of Scrum process breakdown may GASD teams experience during and after sprint planning meetings?
- What forms of Scrum process breakdown may GASD teams experience during and after retrospective meetings?
- What social conditions may lead to Scrum process breakdowns during and after sprint planning meetings?
- What social conditions may lead to Scrum process breakdowns during and after retrospective meetings?

2.9. Chapter Summary

The purpose of the literature review was to validate the research motivation, by demonstrating that while past studies have identified various forms of social challenges impacting the use of Scrum work practices, agile principles, and work practices in GASD and while they have proposed numerous mitigating strategies, not enough studies have to date sought to identify the social conditions leading to the emergence of these challenges. This literature review has also shown that the degree of theorisation on the topic of distributed agile software development is in need of improvement, despite the fact it provides insightful information on how to best conduct distributed agile software development.

3. THEORETICAL FRAMEWORK

The study seeks to understand GASD as a practice and focuses on the work practices of sprint planning and retrospective meetings in globally distributed Scrum. In particular, the study attempts to investigate Scrum process breakdowns which occur when team members engage in such meetings in the GASD context. This study employs a practice perspective, and Bourdieu's Theory of Practice is used to identify and describe the social conditions leading to these Scrum process breakdowns. This chapter elaborates on the theoretical foundation for this study. The chapter justifies a practice perspective to understand the Scrum process breakdowns, as well as the appropriateness of Bourdieu's Theory of Practice. Furthermore, an explanation of how relevant concepts from the Theory of Practice could be applied to explain the Scrum process breakdowns is proposed.

3.1. The Justification for a Practice Perspective

The literature review established that Scrum process breakdowns occur while GASD participants engage in specific work practices. Schultze and Boland (2000) state that the term "practice" is commonly used to distinguish the abstract from the real (as in theory versus practice) through rehearsed and repeated actions. Consequently, whilst studying work practices, one seeks to understand what people do (Singer, Lethbridge, Vinson, & Anquetil, 2010).

A practice perspective postulates that agents behave and interpret the world in certain ways because of shared "practices" which both constrain and enable them (Reckwitz, 2002). Understanding Scrum process breakdown from a practice perspective is particularly useful for this study, as this perspective allows for an understanding of local habits, tacit knowledge and assumptions of social groups (Turner, 1994). A practice perspective was also deemed relevant for this study as it allows for the identification of strategic activities the actors undertake over time while they interact in an organisational context (Corradi, Gherardi, & Verzelloni, 2008).

Also, a practice perspective would allow for an exploration of the complex situation in which work practices are undertaken and the tracing of network of roles within a work setting (Corradi et al., 2008). The unit of analysis then becomes the "practice", making it

possible to demonstrate how any action undertaken by agents is subject to constraints present to the situation, thus leading to Scrum process breakdowns.

3.2. The Justification for Bourdieu's Theory of Practice

Studies adopting a practice perspective have investigated their phenomena of interest using various theoretical lenses like Giddens' structuration theory (e.g. (Orlikowski, 2000)), Activity Theory (e.g. (Nardi, 1996; Kuutti, 1996)), the Communities-of-Practice perspective (Wenger, 1998), or Bourdieu's Theory of Practice (e.g. (Levina, 2006; Levina & Vaast, 2008)). This study employs Bourdieu's Theory of Practice.

Boudieu's Theory of practice was deemed useful to understand Scrum process breakdown as:

(i) It allows us to understand what people do in their daily lives by theorizing about practice

(ii) It highlights the differences among agents and explains how these arise and are negotiated (Levina & Vaast, 2005).

(iii) These breakdowns occur because the GASD team's Scrum work practices deviate from Scrum's best practices. It is thus important to study the actual practices followed by the GASD teams to identify them, implying that the conceptual infrastructure is one of practice.

It is anticipated that the Scrum process breakdowns could be explained by the existence and negotiation of particular differences among the agents. Section 3.4 explains in more detail how concepts from Bourdieu's Theory of Practice can be used to explain the Scrum process breakdowns.

3.3. Theory of Practice

The Theory of Practice, as posited by Bourdieu (1990), is based on the notion that knowledge is constructed as opposed to being recorded. The Theory of Practice challenges the divide between objectively and subjectively driven social research, by incorporating both objectivity and subjectivity to present a theory that represents the practices and experiences of a social group. Bourdieu's approach rests on the premise that objectivism often employs subjective observations which are not explicitly stated while subjectivism often neglects to take into account objective structures and social conditions that contribute to subjective decision-making (Bourdieu, 1992).

Bourdieu posits that the actions of social groups cannot be explained by combining individual behaviours. Instead, these actions are influenced by cultures, traditions, and objective structures within the society (Jenkins, 2002). Bourdieu's work uses the concepts of field, capital, practice and habitus to explain interactions within the social world. These concepts are next outlined and the relationship between them discussed.

3.3.1. Fields

According to Bourdieu (1998), social space can be described as both fields of forces (whose necessity is imposed on agents who engage in it) and fields of struggles (within which agents confront each other according to their position in the structure of the field). It is simultaneously a space of conflict and a space of competition (Bourdieu & Wacquant, 1992). The set of objectives, historical relations between positions anchored in certain forms of power (or capital), and relational configuration inherent to the field, are imposed on agents who enter it (Bourdieu & Wacquant, 1992, pp. 16-17). As mentioned by Bourdieu (1993, pp. 72-73), "in order for a field to function, there have to be stakes and people prepared to play the game, endowed with the habitus that implies knowledge and recognition of the immanent laws of the field".

A field can only exist if the agents within the fields have the dispositions that are necessary to constitute that field and give meaning to it. By participating in the field, agents incorporate into their habitus the proper know how that will allow them to constitute the field (Bourdieu & Wacquant, 1992). Agents within such a social formation manoeuvre, confront each other, and struggle, according to their positions in the structure, in pursuit of desirable resources (e.g. cultural goods, housing, intellectual distinction, social class prestige) (Bourdieu, 1992). These struggles contribute to either conserve or transform the fields's structure (Bourdieu, 1998, pp.32).

3.3.1.1. Fields and Sub-fields

A field can be composed of several sub-fields (or nested fields). An example could be the sub-division of organizations into several departments (sub-fields). Sub-fields have virtual boundaries which define, through specific constraints, whether one can or cannot assume positions within the fields (Hanks, 2005). For instance, a developer can only be employed within a department provided that he is certified (constraint). For this study, a field could relate to the software development project being undertaken, and sub-fields could relate to the different dispersed sites involved in the project.

When agents from various fields engage in collaborative practices such as in GASD project, a joint field of practice often emerges. This joint field emerges as under these circumstances, agents are often in pursuit of common organisational interests (Levina & Vaast, 2005). In addition, according to Abbott (1995), fields of practice may also overlap according to changing practices of agents and the resources they acquire in the process. The GASD context is likely to be characterised by overlapping fields because of the diverse national and organisational contexts separating the team members (Levina & Vaast, 2005).

As mentioned by Hanks (2005), the relationships between the sub-fields are referred to as homology since they share some form of similarity. Homology is both a resemblance between fields or sub-fields and the degree of difference between them (Bourdieu & Wacquant, 1992). Homology occurs because of two reasons (Jenkins, 2002, p. 86):

(1) "It is a reflection of certain commonalities between habitus and practice as they are translated within the differing logics of certain fields"
(2) "It is a consequence of the power of dominant fields, particularly the field of power, to impinge upon weaker fields and structure what occurs within them"

3.3.1.2. Degree of Autonomy of Fields

Fields can also have some degree of autonomy, which varies from one period to another, from one field to another, and from one national institution to another (Bourdieu, 1993). A field has a high degree of autonomy when it "generates its own values and behavioural imperatives that are relatively independent from forces emerging from the economic and political fields" (Naidoo, 2004, p. 458). However, the boundaries of these fields are shifting and imprecise. In that context, a boundary refers to the specific point at which a field no longer impacts on practice (Jenkins, 2002).

3.3.1.3. Field Transformation

People act within the field and thus produce effects upon it (Bourdieu & Wacquant, 1992). A field therefore cannot be considered as static. It changes as practices or power dynamics challenge the field and its boundaries. The dynamics of the field arises out of the struggle of social actors trying to occupy the dominant positions within the field (Bourdieu, 1993). As Bourdieu (1998) puts it "structure is not immutable and the topology that describes a state of the social positions permits a dynamic analysis of the

conservation and transformation of the structure of the active properties' distribution and thus of the social space itself" (Bourdieu, 1998).

A field can be transformed, however, this transformation process is not homogeneous and consistent. Some pockets or sub-sections of the field might be transformed, while others remain unchanged (Webb et al., 2002). In these cases, the field is filled with disagreements over which sub-parts or pockets represent the fields' values.

3.3.2. Habitus

Habitus is a Latin word which refers to habitual or typical conditions particularly to the body (Jenkins, 2002). Habitus relates to the tendencies and pre-dispositions that shape people's conduct, thoughts, feelings and judgments (Bourdieu & Wacquant, 1992). Habitus is internalised in the minds and bodies of agents and further imposes socially accepted behaviour (Hanks, 2005). However as Bourdieu (1990, p.53) posits, habitus is objectively 'regulated' and 'regular' and is not in any way the product of obedience to rules. It can also be collectively orchestrated without being the "product of the organising action of a conductor" (Bourdieu, 1990, p.53). Such is the case as, according to Bourdieu, in any specific situation, rules fail to explain the "messy and strategic nature of social practice" (Sewchurran, 2008).

Thus, there is a relationship between the body and the habitus (Jenkins, 2002). This relationship lies in the embodiment of dispositions in real human beings. This embodiment appears to have three meanings:

(1) Habitus exists inside the mind of actors
(2) Habitus exists because actors engage in practices, interact with each other and their environment
(3) Habitus is inculcated by experience and implicit or explicit learning (Bourdieu, 1993).

Bourdieu also describes habitus as "the generative basis of structured, objectively unified practices" (Bourdieu, 1990, p. vii). Through habitus, schemes which can be objectively consistent with the objective interests of the agents whose habitus it is, can however be generated without having been designed to that end (Bourdieu, 1993, p.76). In particular, practice can be adapted through habitus, to attain certain outcomes without

the agents having a conscious aim of achieving that outcome and without them having a mastery of the operations necessary to attain these outcomes (Bourdieu, 1990, p.53).

Habitus can be perceived as being formed out of history, past experiences, and socialisation processes and includes the embodied and embrained knowledge which agents have acquired (Jarwitz, 2007). The anticipations of habitus (practical hypothesis based on past experience) give a disproportionate weight to early experiences and, unlike scientific estimations, are not corrected after each experiment according to rigorous rules of calculations (Bourdieu, 1990). Past experiences are thus actively present and tend to guarantee 'correctness' of practices and consistancy over time (Bourdieu, 1990).

Habitus constructs the way agents understand the world, their beliefs and their values (Webb, Schirato, & Danaher, 2002). The habitus would then produce individual and collective practices, in accordance with schemes engendered by history (Bourdieu, 1990). Hence, the past survives in the present and, by existing in practices which are structured according to its principles, also perpetuates itself into the future (Bourdieu, 1990). The effect of history is felt through the "objectification in bodies and objectification in institutions" (Bourdieu, 1990, p.57) implying that the history of experience, through practice, is incorporated into both individuals (as habitus) and within institutions (as fields).

Dispositions are durable and transposable (Webb et al., 2002). They are transposable as they can be relevant to other fields or social contexts than the ones in which they were originally created. The durability of dispositions arises because they have been founded in learning from the early years of life. They are also habitual and adjustable to objective conditions of existence (Webb et al., 2002). Habitus is constituted when a set of dispositions meets a particular problem, choice or context. This is known as 'moments of practice' (Webb et al., 2002).

Through habitus, culture is established and communication practices are shaped; these might include gestures, manner of speech, and other verbal and non-verbal embodied actions (Hanks, 2005). As stated by Bourdieu (1993), habitus is a result of the interaction between individuals and the field and it generates and regulates the practices that make up social life.

Strategies engendered by dispositions not only manipulate the established order, but also preserve it by constituting the field (Acciaioli, 1981). However, habitus is not the sharing of a common culture. Instead, each individual disposition (habitus) structurally varies from all the other habitus engendering social interaction (Kelly, 1951). Because of their habitus, agents respond to rules within contexts in various ways. Hence habitus allows for improvisations. However, the responses which emerge are largely determined by where (and who) the agents have been in the culture (Webb et al., 2002).

3.3.2.1. The Arbitrariness of Habitus

Bourdieu sees the practical world as a world of already realised ends (procedures to follow, paths to take) because of the relationship with the habitus (Bourdieu, 1990). In particular, he does not see stimuli as "conditional, conventional triggers acting on condition that they encounter agents conditioned to recognise them" (Bourdieu, 1990, p.53). Also, in contrast to what is regularly observed, agents' aspirations are not directly correlated to the scientifically constructed objective probabilities. Agents do not adjust their aspirations based on their chances of success. In fact, possibilities, impossibilities, freedoms and necessities, opportunities and prohibitions inscribed in the objective conditions inculcate dispositions objectively compatible with these conditions and in a sense pre-adapted to their demands (Bourdieu, 1990). The question of intention, while deciphering the production of practices, can thus be seen as superfluous (Bourdieu, 1990).

Habitus thus operates at a partly conscious level and is entirely arbitrary. Such is the case because whatever values are held by an agent, or whatever practices the agent engages in, are not essential or natural (Webb et al., 2002). But for habitus to function smoothly, the options that the agent chooses from must feel like common sense or inevitable. Habitus (systems, rules, laws, and categories) can only function effectively if the agent forgets about the specific sociocultural contexts of their production. Bourdieu calls this "the forgetting of history which history itself produces" (Bourdieu, 1990).

Habitus is both imposed and imposing (Acciaioli, 1981). Habitus is imposed when individuals or groups feel that no other way of behaving or of achieving diverse tasks is possible other than those prescribed by standards. Other options or the most improbable practices are ruled out (without violence, art or argument) because the agent would see them as unthinkable as they would be incompatible with the objective condition (Bourdieu, 1990). Agents submit to order and refuse what is anyway denied and will the

inevitable (Bourdieu, 1990). In essence, "the conditions in which the individual lives generate dispositions compatible with these conditions (including tastes in art, literature, food, and music)" (Bourdieu, 1990, p. ix). The rules of habitus are thus inscribed on and in agents as if they were human nature. When other rules which are not inscribed in the agents are forced upon them, these are perceived as absurd or even barbaric (Webb et al., 2002).

However, Bourdieu also mentions that, in addition to the practical mode constituted by habitus, there also exists a conscious mode (Lahire, 2011). More specifically, Bourdieu mentions that "it is, of course, never ruled out that the responses of the habitus may be accompanied by a strategic calculation tending to perform in a conscious mode the operation that the habitus performs quite differently, namely an estimation of chances presupposing transformation of the past effect into an expected objective" (Bourdieu, 1990, p. 53). Without being based on any form of calculation, these responses are defined in relation to the objective possibilities inscribed in the present (things to do or not to do, things to say or not to say) in relation to a probable future. Bourdieu uses the term second-order or double-meaning strategies to describe these calculations. Double-meaning strategies occur when ways are employed to make behaviour appear as solely motivated by pure, disinterested respect for the rules by "ostentatiously honouring the values the group honours" (Bourdieu, 1990). In reality, actors might be pursuing values dictated by their economic and political interests (Acciaioli, 1981).

3.3.2.2. Habitus at Collective and Individual Levels

Habitus is embodied at both individual and collective levels (group or class habitus). At the individual level, habitus is acquired through experience and early life socialisation. As the actor proceeds through life and engages in social experiences, there is an adjustment between the habitus and the objective reality.

When the habitus of individuals are exposed to the same fields and the same "logic of action" over an extended period of time (homogeneity of conditions of existence), practices are generated which are "mutually intelligible and immediately adjusted to the structures" without any direct interaction (between agents) or explicit coordination (Bourdieu, 1990, p.58). This gives rise to a class or collective habitus which "enables practices to be objectively harmonised without calculations or conscious reference to a norm" and "without any direct intervention or ... explicit coordination" (Bourdieu, 1990,

p.58). In a differentiated society, the practices of members of the same group are always more and better harmonised than the agents are aware of (Bourdieu, 1990).

However, Bourdieu concedes that class habitus allows for individual difference. Each individual system of disposition contains structural variations and travels along a unique social trajectory. This "brings about a unique integration ... of the experiences statistically common to members of the same class" (Bourdieu, 1990, p. 60). "Personal style", the stamp that marks all practices generated by the same habitus is never more than a deviation in relation to the style of the class. "It relates back to the common style not only by its conformity but also by the difference that makes the 'manner'". (Bourdieu, 1990, p.60).

However, the individual habitus defends itself against the threat of dramatic change by "rejecting information capable of calling into question its accumulated information ... and ... by avoiding exposure to it" (Bourdieu, 1990, p. 60). Habitus would thus favour experiences which would most likely reinforce it. Through unconscious avoidance strategies, it provides itself with a milieu to which it is as pre-adapted as possible and within which the dispositions are reinforced (Bourdieu, 1990). In particular, these avoidance strategies are unconscious.

3.3.3. Capital

Bourdieu (1986) views capital as the specific cultural or social assets that are considered valuable in the field. Based on their respective capital, and their position within the field, actors might have differing levels of power within the field (Webb et al., 2002). Furthermore, by being in a position of power, actors can determine what 'authentic capital' is (Bourdieu, 1999).

3.3.3.1. Forms of Capital

The different forms of capital within a field are symbolic, cultural, and economic (Bourdieu, 1999; Swartz, 1997). Symbolic capital is a particular form of capital which pertains to prestige, honour, or the right to be listened to. Bourdieu (1993) refers to symbolic capital as some form of tacit power that an actor possesses and functions as an authoritative embodiment of cultural value. Cultural capital refers to knowledge, skills, education, and advantages that an agent possesses, thus giving him higher status in society. For instance, children receive cultural capital from their parents who transmit to them the relevant attitude and knowledge required to succeed in the educational system

(Bourdieu, 1986). Economic capital relates to the degree of command that an agent possesses over economic resources (e.g. cash or assets) (Bourdieu, 1986).

3.3.3.2. Fields' Reproduction and Transformation through Capital Distribution

The different forms of capital within the social field can be termed as resources. Capital allows for the transfer and transactions of power (Naidoo, 2004). Capital may be confined to a field, or overlap across fields (Webb et al., 2002). Examples of capital possessed by actors in the dispersed agile software development field could be a Scrum Master certification, or the level of experience of a programmer.

Participants within the field "vie to establish monopoly over the species of capital effective in it" (Bourdieu & Wacquant, 1992, p.17). In order to alter the structure of the field, it is crucial to modify the distribution and relative weight of the relevant forms of capital within it (Bourdieu & Wacquant, 1992). Conflict can thus be derived from the struggles between agents who attempt to either conserve or transform the social space based on their relative positions within the field (Bourdieu, 1990).

The position of agents in the field (either of domination, subordination or equivalence) is based on the effect of the field, the amount of resources they own, and the results of efforts of the agents within that field. Some agents are better players than others within certain field because they already possessed quantities of relevant capital during the process of habitus formation (Grenfell & James, 1998). Conversely, others are disadvantaged.

Upon being aware of their limited amount of capital, new-coming agents to a field often strategize to retain or acquire more capital to improve or preserve their positions in the field (Jenkins, 2002). However, these strategies are limited by the field (Bourdieu, 1993). Such is the case because the field is protected against major disturbances and challenges, which in turn is because these agents would have invested time and effort whilst entering the field according to the fields' own rules (Bourdieu, 1993).

In turn, agents who possess a sufficient amount of relevant capital within a field can dominate that field, whose struggles intensify whenever the relative value of the different kinds of capital is questioned by new-coming agents (Jawitz, 2008). Older members will "try to defend the monopoly and keep out competition" (Bourdieu, 1993, p.72). As pointed out by Bourdieu and Wacquant (1992, p. 98) "it is one and the same thing to

determine what the field is ... and to determine the species of capital that are active in it". A field thus provides the frame of analysis for the study of any aspect of social life.

3.3.3.3. Symbolic Violence and Misrecognition

Bourdieu posits that there are two modes of domination. In the first one, power relations are "mediated by objective institutionalised mechanisms" of which the agents are not consciously aware of. In this form of social formation, agents do not have to endlessly create and restore social relations (e.g. in the education system). In the second mode, dominance relationships are made, unmade and remade through personal interactions. This form of social formation does not inherently contain the mechanisms for its own reproduction and the agents are responsible for its continuous creation (Bourdieu, 1990).

Symbolic violence, in the form of personal authority, relates to the second form of social formation where there is an absence of an "officially declared and institutionally guaranteed delegation" (Boudieu, 1990, p. 129). Under these circumstances, agents with high amount of relevant capital can only sustain their personal authority through actions that reassert this authority through "compliance with values recognised by the group" (Bourdieu, 1990, p. 129). In doing so, their conditions of domination are continuously produced and reproduced and the systems the agents dominate are not allowed to follow their own course.

Agents with relatively high symbolic capital can apply the power against other agents who hold less. In doing so, the holders of high symbolic capital impose their symbolisms and meanings with the purpose of altering the actions of other agents (Jenkins, 2002). Such a process is known as symbolic violence. Symbolic violence occurs when overt violence is collectively disapproved within the field. In this case, symbolic violence, gentle and invisible violence, is then perceived and employed as the only possible way to exercise domination and exploitation (Bourdieu, 1990).

According to Bourdieu, misrecognition occurs when agents are victims of symbolic violence with their own accord or complicity. In essence, when agents are being affected by symbolic violence, they perceive it as legitimate and perceive it as the natural order of things (Bourdieu, 1992; Webb et al., 2002). For instance, an abusive husband with financial and cultural capital might apply symbolic violence on his wife, and she might perceive this as normal. Because the action appears legitimate, the power relations from which it emerges remain hidden (Jenkins, 2002).

Through misrecognition, one can also make sense of the behaviour of some leaders in the field who employ double-meaning strategies claiming that they are behaving in the interest of the field, while in reality they are trying to pursue their own agenda and impose their own values in the field (Webb et al., 2002).

Bourdieu (2000) uses the term *reproduction of symbolic domination* to explain the struggle for capital within a field. In essence, based on their position in the field, agents adjust their expectations of the amount of capital that they are likely to gain. Their position in the field (because of their educational background, social connection, class etc.) imposes on them practical limitations. Consequently, through symbolic domination, agents with less capital tend to have less ambition pertaining to the amount of capital which they can further gain (Bourdieu, 2000).

3.3.4. Practice

Field and habitus are reciprocally constructive and interact under given concrete situations, resulting in the emergence of practices (Bourdieu & Wacquant, 1992; Schultze & Boland, 2000). Habitus orchestrates practice by both harmonising interests and by ensuring that all agents share and adhere to a common consensus on the meaning of the practice (Acciaioli, 1981). It is achieved by a less than conscious process of adjustment of the habitus and the practices of agents to the constraints of the field or social world (Jenkins, 2002).

The link between habitus and practice is not mechanical or deterministic. Because of the habitus, actors are disposed to act in a certain way. Habitus thus provides a basis for practice. Practice is produced because of the encounter between habitus and its disposition, as well as the constraints, demands and opportunities of the social field in which the actor is moving (Bourdieu & Wacquant, 1992).

3.3.4.1. Practical Logic

Practice is neither composed of wholly consciously orchestrated actions, nor of random actions. It is a mixture of both. Because social life is not accomplished on a set of rules, practice has some degree of improvisation. This is termed as practical logic (Bourdieu, 1992). Bourdieu describes practical logic as "a practical mastery of the logic or the imminent necessity of a game – a mastery acquired by mastery of the game, and one which works outside conscious control and discourse" (Bourdieu, 1990, p. 61).

The notion of practical logic is related to the fact that practice is also located in time and space. Whilst analysing practice, it is important to recognise the temporality of its nature because it is emerged in the current time (Jenkins, 2002). One cannot make sense of practice by only looking at the present condition which may seem to have generated it. Nor can one only consider the past conditions which have generated the habitus from which it emerged. Instead, the past social conditions in which the habitus was generated must be related to the present social condition to which it is implemented (Bourdieu, 1990).

Urgency is another property of practice. It is the "product of playing the game and the presence of the future it implies" (Bourdieu, 1990, p. 82). If one stands outside the game as an observer, the urgency, the appeals, and the really lived-in world is swept away (Bourdieu, 1990).

Through the conceptualisation of habitus, Bourdieu attempts not to consider practice as overly deterministic and rather seeks to understand it as "regulated improvisation" (Bourdieu, 1990, p.57). Bourdieu (1992) posits that most actors do not choose to improvise their way in life, but they do not have any other choice. Such improvised actions are in response to the dialectical relationship between a specific situation in a field and habitus (Bourdieu, 1973). Practices are generated by dynamically combining past experience, present situation, and the anticipation of the future consequences of these actions. Practices are not recognised by agents as explicit principles. Instead, they are "embodied corporeally in postures and attitudes and interactionally in the style of strategies whose implementation constitutes practice" (Acclaioli, 1981, p. 31).

The logic of Practice is not that of a logician. By asking more logic from it than what it can give, one would condemn oneself either to extract incoherencies out of it or to enforce coherence upon it (Bourdieu, 1990).

3.3.4.2. Decision-Making

Habitus is the result of an organising action (i.e. a set of outcomes which Bourdieu describes as approximating to structure), a way of being or a habitual state, and a tendency, propensity or inclination (Bourdieu, 1977). Consequently, decision-making appears to be either a reflection of what habitus is doing, an option which emerges from the different possibilities put forward by the habitus, or an illusion, since the principles of its operations are constrained by and derived from the habitus (Jenkins, 2002).

When people engage in decision making about socially competent performances, they do so without making reference to sets of rules. They are incapable of explaining why they decide to behave in certain ways, i.e. habitus derives from the thoughtlessness of habit and habituation (Webb et al., 2002). However, habitus would also operate in relation to the social field. This implies that the same habitus can generate different practices based on the field in which it operates (Jenkins, 2002).

3.3.4.3. Strategies

Habitus generates a series of moves which are objectively organised as strategies but which are not however produced by genuine strategic intention (Bourdieu, 1977). Bourdieu states that even when strategies produced by habitus (to cope with unforeseen and constantly changing situations) appear to be employed to meet some specific ends, they are only apparently determined by the future (Bourdieu, 1977). As stated by Bourdieu (1990), strategies tend to reproduce the structures which produced them.

The manifestations of these strategies can also be to enable agents to improve or defend their positions in relation to other "occupants" of the fields (Naidoo, 2004). This is termed as position taking. "These position-takings are inseparable from the objective positions occupied by the agent as a result of their possessions of a determined quantity of specific capital" (Naidoo, 2004, p. 459).

Consequently, strategies can function as "double-plays" and are objectively organised in such a way that they allow for both the reproduction of the objective structure of "social space" that produced them and for the accumulation of the field specific capital (Bourdieu, 1993). Bourdieu (1973) describes the Theory of Practice as having a dominant circular path where habitus generates practices and practices reproduce the structure (Schultze & Boland, 2000). In this causal loop of generation and reproduction, actors internalise the structure of a field as habitus (see Figure 3.1). Habitus, in turn, generates practices, and practices serve to reproduce and reinforce the structure of the field.

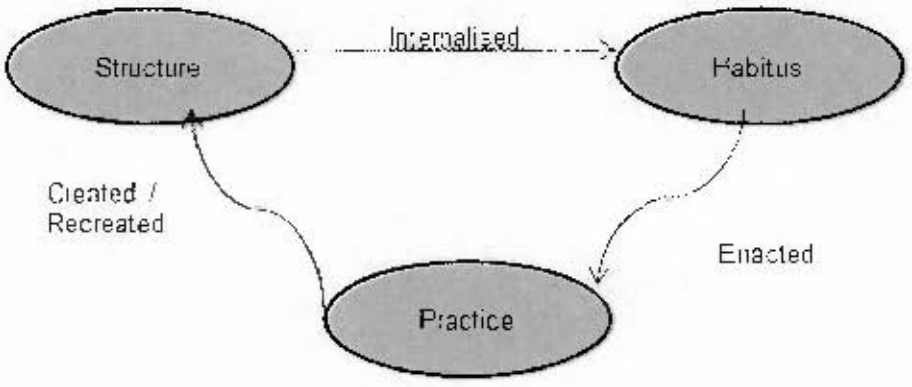


Figure 3-1 - Summary Model of Bourdieu's Theory of Practice (Schultze & Boland, 2000)

3.4. Particularising the Theory of Practice

Various studies employing Bourdieu's Theory of Practice as a lens to understand practice have reported that agents engage in activities (practice) within social spaces which are characterised by overlapping fields of practice (Levina & Vaast, 2005). Bourdieu has employed the metaphor of *games* to characterise these activities (Bourdieu & Wacquant, 1992). In the presence of multiple and nested fields of practice, agents trade capital accumulated in one field for capital in other fields (Levina & Vaast, 2005).

Studies have shown that the agents' interests and competencies would in turn be shaped by each of the fields in which they engage (Levina & Vaast, 2005). In addition, agents can be both united or divided as a result of these overlapping fields of practice. An example of agents being divided can be seen in Levina and Vaast (2008), where the researchers specifically studied collaboration in the offshore software development scene, and reported that overlapping fields cause agents to be separated by the lack of shared symbolic, social, economic, and intellectual capital, which further introduces power dynamics which hinder collaboration. In this particular case, the divide occurred across overlapping fields, based on differing interests and practices (Levina & Vaast, 2005).

Drawing on evidence from these past studies on the impact of overlapping fields of practice, this study seeks to describe the social conditions giving rise to Scrum process breakdowns experienced during and after sprint planning and retrospective meetings by particularly focusing on the divide which arises as a result of overlapping and nested fields (sub-fields) while participants engage in GASD projects.

Conflicts can also arise within a field of practice when agents compete for capital (Bourdieu, 1990). In addition to being part of several overlapping fields, GASD participants also belong to a common joint field when they engage in a common GASD project. Specific forms of symbolic, economic, social, and cultural capital are deemed relevant within this joint GASD field. That field also has specific forms of habitus through which the laws of the field are recognized. It is anticipated that Scrum process breakdowns occurring during and after sprint planning and retrospective meetings can be explained by the internal conflict occurring within the GASD project field while participants compete for capital within the joint field.

The purpose of a retrospective meeting is to enhance the practices being followed during a sprint, based on the team's ideas on how to resolve the issues experienced during the previous iteration (Schuh, 2005). Based on the agile principles, the Scrum team should be empowered to propose and implement such ideas (Koch, 2005). However, agents share specific positions and possess specific capital within the fields, and improvement suggested by team members could be perceived as schemes to enhance their positions within the overlapping, joint and nested fields. Not all participants would be inclined to change the process, and failed retrospective meetings could be attributed to inability to transform the field. Bourdieu (1990) states that conflict arises when agents attempt to transform or conserve the social space based on their relative positions in the field. A Scrum process breakdown experienced during and after retrospective meeting could thus be attributed to the struggles between the agents in the fields and their attempts to maintain or enhance their positions.

3.5. Chapter Summary

The purpose of this chapter was to rationalise the need for a practice perspective in order to answer the proposed research questions. In addition, the use of Bourdieu's Theory of Practice to explain the forms of social challenges experienced while engaging in GASD using Scrum was warranted. A detailed overview of Bourdieu's theory was also provided along with a description of how Bourdieu's various concepts could be useful to interpret the data and answer the research questions in order to make a valid contribution to practice and knowledge.

4. RESEARCH METHODOLOGY

This chapter presents the methodology employed to undertake this research study given the primary and secondary research questions. First, the case design section in which discussions on the selection of the relevant empirical situations, the unit of analysis, the selection of informants and other data sources, and how validity and reliability was ensured, are provided. The two case studies are then described with particular focus on where, when, and how the data was collected in each case. The projects investigated during the case studies are also presented in this section. A detailed account of the data analysis process is also provided. The final section handles issues of access, privacy, confidentiality and ethics. The study is empirical, positivist and qualitative in nature.

4.1. The Case Study Research Method

This study employed a case study research method. Empirical in nature, this method is used to investigate contemporary phenomena within real-life contexts, especially when the boundaries between the phenomena and the context are not clearly evident (Yin, 2003). Case study research method thus serves to capture, document, and theorise about practitioners' knowledge and experiences (Benbasat et al., 1987). The choice of the case study method also particularly suits the format of the research question being posed in this study, namely "what", allowing for an understanding of the complexities of the processes being followed (Benbasat et al., 1987).

4.1.1. Case Study for Theory Building

As previously mentioned (see Chapter One), the purpose of this study is to propose a level 4 (Llewelyn, 2003), mid-range, descriptive theory (Gregor, 2006), to describe the social conditions under which Scrum process breakdowns are experienced by GASD team members during and after sprint planning and retrospective meeting. Case study research was found useful to fulfil this purpose as the method has proved to be particularly suited for theory building (Van de Ven, 1989). In particular, as recommended by Voss, Tsikriktsis and Frohlich (2002), while seeking to theorise about the Scrum process breakdowns during GASD, the case study has been directed at:

- Identifying and describing the key variables for the theory
- Identifying how the variables are related to each other
- Identifying why these relationships exist

4.2. Case Study Design

The case study was designed and executed to gather rich data and uncover local habits, tacit knowledge of the social group under study, and other forms of assumption (Turner, 1994). In addition, the case study design follows the recommendations of Voss et al. (2002) who suggest the following steps:

(1) Identification of the research framework, constructs, and questions
(2) Case selection
(3) Developing research instruments and protocols
(4) Conducting the field research
(5) Ensuring reliability and validity in case research
(6) Data analysis

4.2.1. STEP 1: Identification of the Research Framework, Constructs, and Questions

In line with Voss et al. (2002), the first step in designing a case study was to identify the research question. An initial conceptual framework (see Figure 3.1) was also employed to provide an initial view and "force the researcher to think carefully and selectively about the constructs and variables to be included in the study" (Voss et al., 2002, p. 199)

4.2.2. STEP 2: Case Selection

The following three important aspects of case study design were considered in the case selection: (1) the number of cases to be included, (2) the unit of analysis, and (3) the case(s) selection and sampling. These are further elaborated below.

4.2.2.1. Decision on the Number of Cases

Ethnographic case studies are ideal for obtaining a deep understanding of a research question and often take the form of longitudinal studies (Voss et al., 2002). A single in-depth case study would have been ideal to answer the research questions. However, due to resource limitations (time and financial constraints), it was not possible to conduct a single case study; instead the research opted for multiple case studies of shorter time frames. This goes in line with Voss et al. (2002)'s recommendation to use multiple case studies when resources are constrained as a means to leverage external validity and prevent observer bias. Benbasat et al. (1987) further confirm that multiple case studies can be used for theory building, which suits the purpose of the study.

Bonoma (1985) proposed a four-stage process for the application of case study research. The four stages are "drift", "design", "prediction", and "disconfirmation" (Bonoma, 1985, p. 204). The case studies conducted for this research are in line with the "drift" and "design" stages. The "prediction" and "disconfirmation" stages were not executed due to time and financial constraints. However, discussion on how these stages should be applied to further validate the theory being formulated at the end of the study has been provided in Sections 9.4.3 and 9.6.

During the first case study, the researcher focused on learning about concepts, context, and jargon of the phenomenon of interest, i.e. distributed Scrum as it occurred during GASD. This case study, allowed for further reflection on and refinement of the interview questions (Bonoma, 1985). After analysis of the empirical data obtained from the first case study, preliminary descriptions of the social conditions under which the Scrum process breakdowns were seen to occur was also proposed.

The empirical data obtained from the second case study served to confirm the Scrum process breakdowns identified in the first case. It also served to assess and refine the description of the social conditions under which they were seen to occur. This was in line with Bonoma (1985)'s "design" stage. This stage requires the researcher to go back to the "drift" stage thinking mode if the early forms of conceptualisation identified in the first case study are not confirmed by the new data or if better forms of conceptualisations are identified (Bonoma, 1985).

Both the "prediction" and "disconfirmation" stages were not executed but are crucial steps to fully test the generalizability of a theory. It is thus proposed that these stages be undertaken as future research (see Section 9.6).

4.2.2.2. The Case(s) Selection and Sampling

The selection of cases is an important but difficult aspect of case study research (Yin, 2003; Benbasat et al., 1987). A mixture of opportunistic and purposeful sampling was employed in this study. The cases were partly selected based on convenience as well as the GASD project team's availability and willingness to participate in the study. However, care was taken to ensure that the characteristics of these GASD project teams were appropriate to answer the research questions in a credible way. This was achieved through purposeful sampling. Appendix 11 summarises the purposeful sampling dimensions chosen for this study. These dimensions, chosen based on the definitions of

specific theoretical constructs from the Theory of Practice relevant to the study, pertained to:

- Number of software development sites
- Degree of cultural diversity at various sites
- Agile methodology employed
- Past software development methodology used by the team
- GASD format (e.g. offshore development, outsourcing)
- Management structure
- Level of experience of team members with software project
- Team members' / stakeholders' monetary investment in the project
- Level of experience of team members with Scrum
- Level of experience of team members with various communities of practice within the software development industry

The second case was selected to achieve literal replication (Miles & Huberman, 1994) as it allowed for patterns to be replicated across both cases (pattern matching), thus confirming the relationship between the constructs and increasing validity (Stuart, McCutcheon, Handfield, McLachlin, & Samson, 2002). The use of literal replication also matched the two stages of case study design (Bonoma, 1985) employed in this study, i.e. drift and design stages. Further details on how the respondents were selected are provided in Section 4.5.3.

- **Identification of the Organisation for Case 1**

Identifying appropriate cases was difficult. In SA, organisations employing agile methodologies were mostly collocated and were thus inappropriate for the study. Fortunately, the researcher was introduced to a certified Scrum trainer for the SA region who provided the name of an organisation involved in GASD. During the preliminary negotiations with the organisation, the researcher ascertained that the GASD team to be investigated matched the criteria specified in Appendix 11. A formal request was then made to the organisation's CEO to undertake the case study. The aim and purpose of the research were clearly explained to the CEO, along with the proposed time frame for the study. He also requested access to the main interview questions in order to review them. A memorandum of understanding was signed to formally guarantee that no information provided would be divulged to any external party.

Prior to beginning the first case study, the researcher attended the Scrum Master Certification course provided the Scrum trainer who recommended the organisation for Case 1. Some team members from the GASD team to be studied in Case 1 also attended the same course with the same Scrum trainer. Hence, by attending this course, the researcher obtained an insight on the views and perceptions shared by these GASD team members on agile methodologies and Scrum. At the end of the course, the researcher undertook an examination and obtained a *Certified Scrum Master* title from the Scrum Alliance Group. Eventually, the researcher was introduced to the GASD team, both on the SA and the Brazilian side, and the first case study could begin.

- **Identification of the Organisation for Case 2**

The second organisation case study was identified through the Cape IT Initiative (CiTi), a non-governmental organisation whose aim is to promote the IT industry in Cape Town. Since the director of CiTi collaborated with various international IT organisations, she was in contact with one of the directors of an Indian software and services organisation which acted as an IT service provider to various South African companies. This organisation employed distributed Scrum to manage their software development projects.

Having been given the contact details of the organisation's director in India, the researcher started negotiating access to the organisation and to one of its GASD teams in India. Similarly to C1, she ensured that the team possessed the adequate characteristics as described in Appendix 11. In addition to several email communications, a tele-conference session was held with the director to ascertain that the nature of the GASD team was appropriate and to negotiate the dates during which the case study would be conducted in India. Care was taken to ensure that the case study would last an entire sprint. A memorandum of understanding was also signed to guarantee that no information would be divulged to any external party.

4.2.3. STEP 3: Research Instruments and Protocol for Data Collection

Voss et al. (2002) recommended that the case study protocol not only include information about the research instruments, but also from whom and from where the different types of information was sought, as well as the subject covered and core research questions posed during the interview. Hence, details about how these specific elements, which were employed in the study, are provided below.

To provide a wider scope of coverage and a fuller picture of the phenomena being studied (Bonoma, 1985), data was collected through semi-structured interviews, direct observation, documentation and field notes. This was particularly useful for triangulation across the various data sources. This was beneficial since it provided multiple perspectives on issues around Scrum process breakdowns experienced by the GASD participants, supplied more information on emerging concepts, allowed for cross-checking, and yielded stronger substantiation of constructs (Glaser & Strauss, 1967). In line with Voss et al. (2002) triangulation was achieved by identifying discrepancies in the responses of the various GASD participants, what was being observed during the meetings and daily interactions, and/or the documentations, and by seeking clarification on these. It is acknowledged that the respondent sample size was small. Hence, in order to compensate for this limitation, efforts were made to gather as much information as possible from all the data sources.

4.2.3.1. Semi-Structured Interviews

Semi-structured interviews were the primary source of information for this study. A version of the interview protocol employed during the case studies is given in Appendix 12. The list of questions in the protocol does not represent the finite set of questions which were asked during the case studies. More detailed questions were asked based on the responses obtained from the participants. In addition, through the field notes which were taken on a daily basis (details provided in subsequent sub-sections), gaps in the researcher's understanding of the GASD team members' work practices were identified and interview questions were formulated to further investigate them. Thus, the interview protocol only served as a general guide for the conversation.

The questions presented in Appendix 12 were not asked in a systematic and sequential manner. Based on their roles in the team, participants were asked questions which they were more likely to provide useful and informed answers. For instance, team management-related questions were mostly posed to participants having a management related role, or questions on testing were mostly posed to testers. However, other participants were also questioned (to a lesser extent) on these topics in order to consolidate the information obtained from the other respondents.

Semi-structured interviews allowed for access to the participants' interpretation regarding the actions and events which had taken or were taking place (Walsham, 2002b). Interviews were open-ended and assumed a conversational manner. Each

respondent was interviewed for at least an hour and care was taken to retain the balance between excessive passivity and over-direction. Furthermore, by not being excessively passive, an interest in the respondent's answers was demonstrated (Walsham, 2002b). The previous responses from the other respondents were also kept in mind while conducting the interviews and the researcher was also "aware of the significance of what [was] left unsaid as well as what [was] said" (Voss et al., 2002, p. 207). This allowed for information that might have already been obtained to be corroborated. Special care was taken to elicit the respondent's views and experiences in his or her own terms rather than to collect data that was simply a choice among pre-established response categories (Kaplan & Maxwell, 1994).

The gathering of information in an organisational setting is challenging due to confidentiality issues. Also, since the researcher and the interviewees did not know each other prior to the interview sessions, some trust concerns could have impacted on the data collection process. Such a situation would have hindered the respondents from being open in their responses and might even interfere with their usual behaviour (Webb, Campbell, Schwartz, & Sechrest, 1966). As such, care was taken while formulating the questions to ensure that they were as non-personal as possible. Also, since formal approval from management and the CEOs was obtained, the respondents felt more at ease talking without fear of divulging confidential information.

To capture everything that was said and to spot salient points while transcribing, a tape recorder was used. However, it was not taken as a substitute for listening closely throughout the course of the interview (Yin, 2003). Notes were taken during the interviews about important points on which further discussion was required.

All interviews were conducted in English. Language being sometimes ambiguous (Fontana & Frey, 2000), it was at times difficult to convey the right meaning while asking a question. The same applied while obtaining responses from the interviewee. This was particularly felt while interviewing the Brazilian and Indian participants. In these situations, the use of a recording device was useful. It allowed the researcher to go over the responses several times to better grasp the idea being put forward by the respondents.

Table 4.1 details the interviews conducted during the case studies. A total of 16 individual interviews and one group interview were conducted. The case number, site,

job title, duration and mode of interview have been provided. As can be seen in Table 4.1, the respondents were firstly selected according to their job titles and levels of experience within the teams in anticipation that they would provide varying perspectives on the work practice of the team. Secondly, the respondents were selected according to the key roles within the Scrum team as it was anticipated that the various roles would have different goals and interests within the Scrum process and engage in varying work practices. Thirdly, snowballing was employed, whereby whenever one respondent mentioned that one particular team member or stakeholder had key knowledge on a specific topic, that person was also interviewed.

As can be seen in Table 4.1, in the second case all the respondents belonged to the offshore site. This is the case as the organisation explicitly requested, for confidentiality reasons, that the researcher has no contact with the customers at the other remote site. It is acknowledged that the data for the second case could have been biased. However, care was taken to limit the negative impact of the situation. For instance, during observation sessions of the meetings between the offshore and onshore team members, the researcher carefully noted down her impressions of the behaviour and responses of the onshore participants. The nature of the interactions between the onshore and offshore team members during these meetings was an invaluable source of information on the social dynamics taking place within that particular GASD Team. The field notes were also further corroborated by comments made by the Indian team members during interviews. These team members were also thoroughly questioned on the roles and behaviour of onshore team members. Their responses were corroborated and gave an indication of where these remote participants fitted in the development process. The researcher also attended a focus group with several agile developers from other teams during the second case study, to discuss their distributed agile software development. Details about when the case studies were conducted and how long they lasted will be provided in the case study description sub-sections.

Case No.	Site	Job Title	Interview Duration	Level of Experience in team	Mode of Interview
1	Cape Town, SA	COO	1 hour	2 years	Face-to-Face
1	Cape Town, SA	Product Owner	1 hour 10 mins	1 year 3 months	Face-to-Face
1	Cape Town, SA	Scrum Master	1 hour 15 mins	1 year 6 months	Face-to-Face
1	Cape Town, SA	Technical Architect	50 mins	3 months	Face-to-Face
1	Cape Town, SA	CEO	1 hour 10 mins	3 years	Face-to-Face
1	Sao Paulo, Brazil	Lead Technical Architect	45 mins	2 years	Conference call
1	Sao Paulo, Brazil	Front End Developer	50 mins	1 year 8 months	Video Conference
2	Pune, India	Project Manager + Delivery Unit Head + CEO	1 hour	N/A	Group interview
2	Pune, India	Project Manager	1 hour 5 mins	2 years 6 months (5 years with the organisation)	Face to-face
2	Pune, India	Delivery Unit Head	1 hour	5 years (with organisation as a whole)	Face-to face
2	Pune, India	Technical Specialist	1 hour 15 mins	9 months	Face-to-Face
2	Pune, India	Developer	45 mins	2 years	Face-to-Face
2	Pune, India	Team Leader (Development)	1 hour 10 mins	2 years 6 months	Face-to-Face
2	Pune, India	Tester	1 hour	2 years 6 months	Face-to-Face
2	Pune, India	Tester	40 mins	<6 months	Face-to-Face
2	Pune, India	Junior Developer	50 mins	1 year	Face-to-Face
2	Pune, India	Developer	55 mins	2 weeks	Face-to-Face

Table 4-1 - Interviews Sessions

4.2.3.2. Direct Observation

Direct observation involves, as much as possible, the formulation of a thoughtful, objective and precise record of group interactions and behaviours (Babbie, 1992), through "seeing" and "listening" (Taylor-Powell & Steele, 1996). Care was taken not to be confined to looking at pre-set items, and instead to see things within the participant's context. Special attention was placed on "what was not happening" as well as on "what was happening". In addition to the Sprint planning and retrospective meetings, daily scrum meetings, as well as the general day-to-day of activities of the team members were also observed, as a means to capture evidence of consequences of the Scrum process breakdown. A total of 28 Scrum meetings, sprint planning, and retrospective meetings were observed as shown in Table 4.2.

Case No	Meeting Type	Number of observations
1	Scrum meeting	8
1	Sprint Planning 1	1
1	Sprint Planning 2	1
1	Retrospective meeting	1
2	Pre-Scrum meeting	4
2	Meeting to discuss way forward for team	1
2	Sprint Planning 1	1
2	Sprint Planning 2	1
2	Scrum meeting	10

Table 4-2 - Observation Sessions

4.2.3.3. Documentation

The documents reviewed during the case studies include user stories, task boards, bug reports, specifications, burn-down charts, and emails exchanged between the team members and the customers. Access to the documents was granted by the COO (in C1) and the project manager (in C2). The documents were either stored in central repositories (e.g. wiki) or email databases.

The documents were primarily used to augment evidence from other sources. If the documentary evidence was contradictory rather than corroboratory, the topic was further investigated. All required documents were looked at in a systematic way.

4.2.3.4. Field Notes

Field notes are defined as "running commentaries" about the events unfolding during a case study (Eisenhardt, 1989). Field notes were recorded into a notebook computer on a daily basis including contextual data, personal comments and immediate interpretation of events (Kirk & Miller, 1986). Relevant information gathered during informal conversations with team members was also described in narrative descriptions.

The field notes were written during the day while the researcher was sitting in the office with the other team members. These field notes were further examined during the evenings and formed the basis of further investigation (interviews, further observation or documentation) on the next day. A sample field note is provided in Appendix 13.

Validation is a key step during any research and was ensured by limiting the bias and the researcher's influence on the environmental setting. In essence, bias is introduced as the researcher may see what he or she wants to see (Taylor-Powell & Steele, 1996). In addition, even though passive observers do not participate in the group they are studying, they still need to be aware of what influences the act of observing might have on the participants' behaviour (Montgomery & Duck, 1991). Therefore, while recording the field notes, care was taken to ensure that the participants did not feel uncomfortable by the fact that they were being observed and subsequently modified their behaviour. During the observation of the various meetings, the researcher always sat in an undisruptive spot and allowed the participants to engage in their usual meeting routine. These meetings were also recorded and later transcribed to corroborate the field notes. Prior permission to observe these meetings were also obtained.

4.2.4. STEP 4: Ensuring Reliability and Validity of the Case Study

While conducting case study research, one needs to pay particular attention to reliability and validity related issues. *Construct Validity and Internal Validity* are the two dimensions of validity.

Construct validity refers to the degree to which correct operational measures are established for a concept (Voss et al., 2002). Given Voss et al. (2002)'s recommendation, care was taken to achieve construct validity in this study by:

- Confirming whatever relationships predicted between constructs through further observations within and across the cases.
- Gathering data through multiple sources, including semi-structured interviews, direct observations, documentations, and field notes
- Checking if constructs can be differentiated from each other

The tactics to ensure construct validity were employed during the data collection phase. Internal validity was achieved during data analysis. Internal validity relates to the extent to which causal relationships between constructs can be identified (as distinguished from spurious relationships) (Yin, 2003). The techniques employed to achieve internal validity will be described in the data analysis subsection (Section 4.8).

Reliability refers to the extent to which the operations undertaken during the case study can be replicated and will yield the same results (Yin, 2003). This has been achieved through a careful description of the case study protocol. In the following subsections, contextual information is provided about the three case studies conducted during this study.

4.3. Case Study 1: SA and Brazil

Data for the first case study was collected over a period of four weeks between March 2009 and April 2009 in Cape Town. When data collection started, two sprints had already been completed. During the first week of data collection, the team was in a transition phase between the second and the third sprint. The data collection lasted throughout the duration of the third sprint; this allowed the researcher to observe the sprint planning meetings, daily Scrum meetings and retrospective meetings. During the first week, the researcher undertook interviews, observed and familiarised herself with the GASD team's work environment and practices on a daily basis to be better prepared for in-depth observations when the sprint started. A summary of the details about the first case study is provided in Table 4.3.

Start date	March 2009
End date	April 2009
Case study duration	Four Weeks
Sprint duration	Three Weeks
Number of meetings observed	11
Number of interviews conducted	7
Number of working days spent in the field with the team members	19
Project	MKX Knowledge Management Software Application

Table 4-3 - Summary of Case Study 1

The first case study (C1) was undertaken in one GASD team within an IT organisation. The organisation formed part of a group of media companies (HIM Group) dispersed all over the world. The HIM group's business strategy was centred on creating media content, building brand names around it, and managing the platforms distributing the content. The organisational activities revolved around research and media product development. All the subsidiaries in the group worked in silos and were often in competition with each other. Hence, the organisation was striving to create synergies amongst the subsidiaries through the evangelisation of the adoption of knowledge management practices in the group. For instance, the organisation was promoting software application sharing, instead of re-inventing the wheel by re-writing similar software applications for different subsidiaries in the group.

4.3.1. The C1 Team

The team within which the case study was conducted can be described as a multi-disciplinary global product development team. The team was comprised of Internet analysts, product specialists and software engineers and described itself as *"a think tank and execution arm of the Internet division within the group"*. The team was split across two countries: SA and Brazil. Brazil (Sao Paulo) was chosen as one of the sites for the team because of the high project workload demand at this location. SA (Cape Town) was chosen as a site as it is physically centrally located in relation to the time zone.

At the time of the case study, the team was composed of three members located in Sao Paulo and four members in Cape Town, and operated under the leadership of the Chief

Executive Officer (CEO) and the Chief Operations Officer (COO). A background description of each respondent in terms of gender, position in the team, years of service, and location is provided in Table 4.4.

The team's composition was rather unstable at the time of the study. As can be seen in Table 4.4, the CEO was the only respondent who had been part of the team since its inception. According to the CLO, the fluctuation in the team's composition was mainly due to high project workload and staff shortage. Software developers were thus shifted across project teams based on workload demands and staff availability.

One developer at the South African site and one lead developer at the Brazilian site were not interviewed. The software developer in Cape Town was not interviewed as he was not identified as a key informant by any of his fellow team members during the snowballing procedure. This could be because he had only been part of the team for one week at the time. The lead developer at the Brazilian site, even though he would have been an interesting informant, was not interviewed as he was often not in the office during the times of the field visits due to time zones difference between Sao Paulo and Cape Town. Several attempts had been made to schedule video conference sessions with him but it proved unfruitful. Instead, information about his role and purpose was sought from his colleagues. Though not ideal, this process mitigated the limitations of not having interviewed him to some extent.

Resp. Pseudo.	Gender	Job Title	Years of Service with Team	Location
C1.R1	Male	COO	2 years	Cape Town, SA
C1.R2	Male	Product Owner	3 months	Cape Town, SA
C1.R3	Male	Scrum Master	1 Year 6 months	Cape Town, SA
C1.R4	Male	Technical Architect	2.5 months	Cape Town, SA
C1.R5	Male	CEO	Since the creation of the team (3 years)	Cape Town, SA
C1.R6	Male	Lead Technical Architect	1 year 6 months	Sao Paulo, Brazil
C1.R7	Female	Front End Developer	1 year 8 months	Sao Paulo, Brazil

Table 4-4 – Description of C1 Respondents

Figure 4.1 outlines the GASD team at C1.

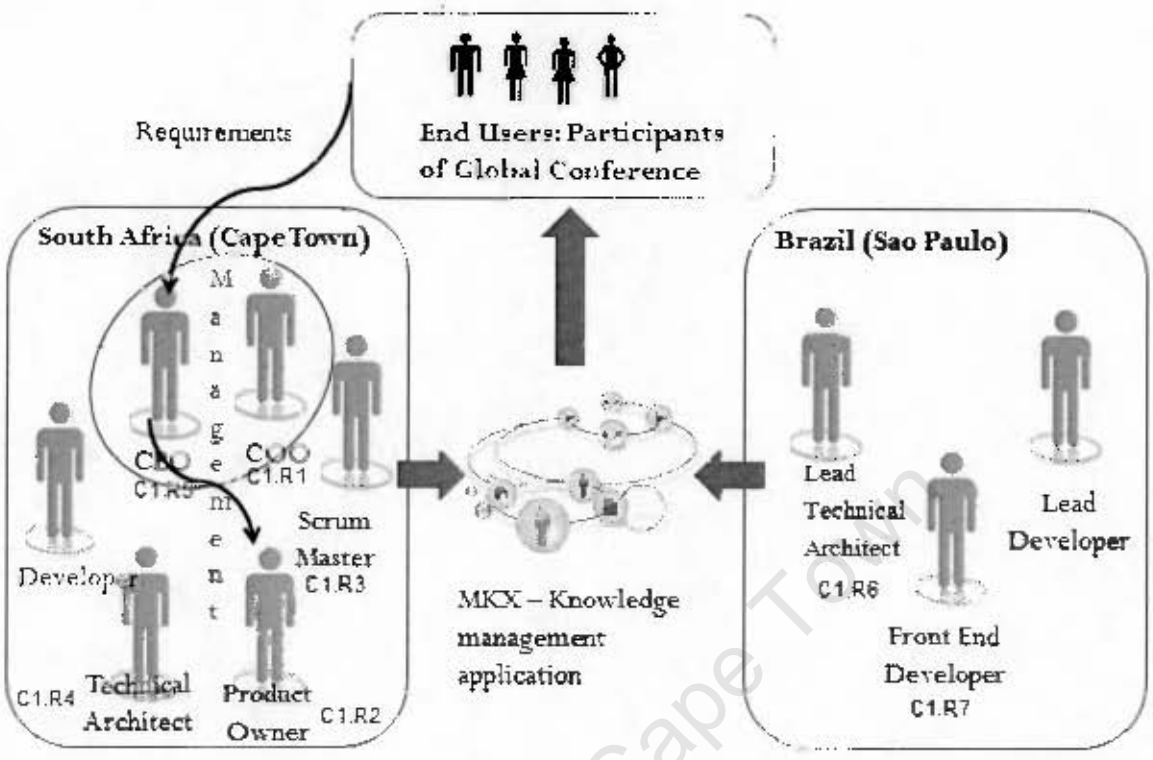


Figure 4-1 - Globally Distributed Team at C1

4.3.2. The C1 Project

In line with the vision of knowledge sharing across the group, the team was involved in the development of a knowledge management tool (MKX) aimed at leveraging the extent to which individuals in the group communicate with each other. The participants explained that the application was initially developed using the Waterfall approach, which proved inadequate for the project nature and setting. Consequently, the Scrum methodology was adopted. The MKX application was mainly developed using PHP. The MKX application was already operational. However, the project was still on going since new features were added whenever the need arose. Details about the project are provided in Table 4.5.

Project Name	MKX Knowledge Management Software Application Project
Project Purpose	Facilitate and enhance knowledge creation, capturing, representation, sharing and distribution between the employees of the HIM group. Through the use of the MKX application, employees of the group can form networks and sub-communities. The application also aims to facilitate communication and collaboration between members of these networks, especially prior and during events like conferences organised by the HIM group.
Project Initiator	CEO
Project Start Date	February 2008
Methodology employed	Waterfall Approach: February 2008 – December 2008 Scrum: January 2009 – time of case study
Team composition	GASD Team composition had high fluctuation rate since beginning of project
Programming Language	Front-end: Javascript Back-end: PHP
Architecture	Model View Controller (MVC)

Table 4-5 - MKX Application Project Description

4.3.3. The C1 Work Setting

To describe the work setting of the C1 GASD Team, the "stage" analogy proposed by Myers and Newman (2007) has been used. In particular, Myers and Newman (2007) posit that it is important to carefully describe this stage as part of the research process.

The overall office in Cape Town was composed of four rooms. The first and largest one could fit about ten people. It was a common office space where the team engaged in their daily development tasks. The second and third rooms were personal offices for the CEO and COO. The side offices were closed by glass panes; the team members, the CEO and the COO could all see each other. A coffee machine was available and employees could go for coffee breaks at any time. The fourth room was the conference room. All interviews were conducted in the conference room. The conference room was spacious and had a large flat screen TV for video conferences. A large oval table was at the centre of the room with chairs around it. The venue was not necessarily tidy and conveyed a rather casual atmosphere. A tele-conference phone was placed at the centre of the table

as well as a Nintendo Wii™. On the wall, there was a large framed picture of a Brazilian Carnival dancer wearing a pink outfit. There were white boards on the walls with Unified Modelling Language (UML) diagrams sketched on them as well as information detailing some of the requirements.

4.4. Case Study 2: India and SA

Data collection for the second case study lasted four weeks and was undertaken between May 2010 and June 2010 in Pune (India). At the time of the case study, the GASD team was working on the migration of an online gaming application to a new platform (Colin Application Project) and eight sprints had already been completed. The case study lasted throughout the duration of the ninth sprint during which the researcher observed the sprint planning meetings, daily scrum meetings, and a retrospective meeting, along with the daily activities of the GASD team members. The researcher entered the field three days prior to the beginning of the sprint and exited three days after the end of the sprint. A summary of the details about the second case study is provided in Table 4.6.

Start date	May 2010
End date	June 2010
Case study duration	Four Weeks
Sprint duration	Three Weeks
Number of meetings observed	17
Number of interviews conducted	10
Number of working days spent in the field	20
Project	Colin Application Project

Table 4-6 - Summary of Case Study 2

The organisation for the second case study (C2) was a branch of a global software and services organisation operating in 18 countries around the world including India, United States (US), United Kingdom (UK), Germany, Sweden, SA, and Singapore. The branch located in India delivered services for mission-critical applications, enterprise applications, e-business, and Business Process Outsourcing (BPO) services. With a Capability Maturity Model (CMM) Level 5 certification, it provided expertise that spanned across Retail and Distribution, Banking, Financial Services and Insurance, Healthcare, and Life Sciences, Manufacturing, Energy and Utilities. The customer base for O2 was composed of over 400 clients with whom they had long term relationships. On average O2 retained its customers for about five years. Some customers even stayed on board for over 12 years. O2 had over 5000 employees.

As illustrated in Figure 4.2, the organisation was organised into three units: Practice, Sales and Delivery, whereby each Delivery Unit met the need of one specific customer. A Delivery Unit was dedicated to one customer and could execute several projects for that specific customer over time. The Sales Unit interacted with both the customers and the Delivery Units. The Practice Unit was responsible for deciding which services should be sold to the customers. The decisions were based on feedback from the Sales Unit on what the customers were asking for. The Practice Unit interacted with both the sales and the Delivery Units. The Practice Unit was also responsible for observing industry experts like Gartner in order to determine the trends in the IT industry (e.g. agile or cloud computing). These trends were then proposed to the customers as service offerings. Examples of service offerings might have been the creation of an application for a retail customer or the maintenance of an online application for a bank. Consequently, the Practice Unit created solutions and these solutions were proposed to the customers. When the customers placed orders, these orders were executed by the Delivery Units (See Figure 4.2).

Figure 4-2 – Modus Operando for O2

4.4.1. The C2 Team

The C2 team operated in Pune and Durban. The Pune team members belonged to a Delivery unit and the SA team members were Customers of C2. The project was thus

executed by the Delivery Unit and the Customers who consequently outsourced part of the development work of their project to India. In C2, the terms "team members" or "team" refer to the whole development team spread across India and South Africa.

All team members were highly qualified and comprised an on-site coordinator, a project manager / Scrum Master, a lead tester, testers, a business analyst / Product Owner, a technical Scrum Master, a team leader, software developers, a technical specialist and a project manager. Figure 4.3 illustrates the distribution of team members across the sites.

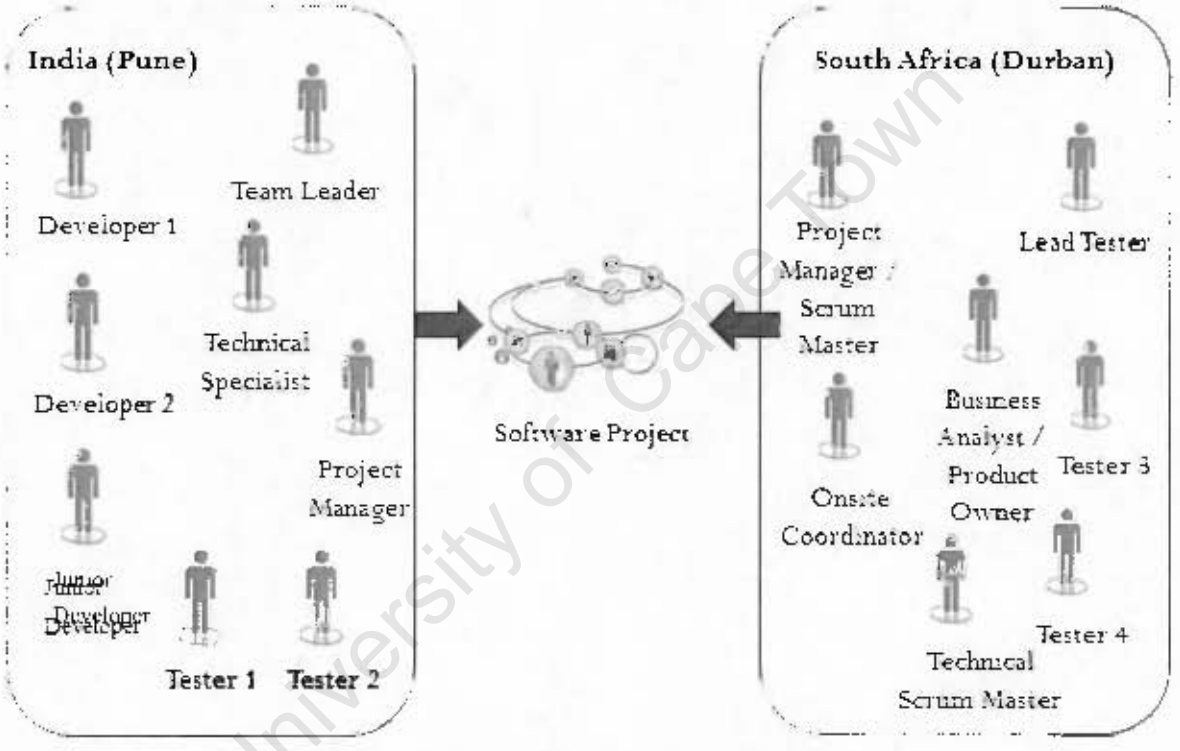


Figure 4-3 - Globally Distributed Team at C2

The on-site coordinator was responsible for facilitating the deployment of releases at the SA site. He also facilitated the communication and collaboration between the dispersed team members. His was of Indian nationality and he was selected for the position because of his cultural affiliation with both the SA and the Indian team members. The team members at C2 retained the job title which they held prior to the implementation of Scrum within the team.

A background description of each respondent in terms of gender, position in the team and years of service in the team are provided in Table 4.7. Respondent C2.R9 in Table 4.7 has not been represented in Figure 4.3 as he was the Delivery Unit Head and not part of the C2 team.

Resp. Pseudo	Gender	Job Title	Years of Service with Team
C2.R1	Female	Tester	< 6 months
C2.R2	Male	Tester	2 years and 6 months
C2.R3	Male	Team Leader	2 years and 6 months
C2.R4	Male	Developer	2 years
C2.R5	Male	Junior Developer	1 year
C2.R6	Male	Technical Specialist	9 months
C2.R7	Female	Developer	2 weeks
C2.R8	Male	Project Manager	2 years 6 months (5 years with organisation)
C2.R9	Male	Delivery Unit Head	Not part of team (5 years with organisation)

Table 4-7 – Description of C2 Respondents

4.4.2. The C2 Project

The online gaming software application on which the team was working was initially developed by an Israeli company. The SA organisation for which O2 was acting as a service provider, had initially outsourced the development of the application to the Israeli company who developed it from scratch. The project had two parts: (1) a "skeleton" application to be used by the casino affiliates and (2) a "casper" application to be used for the management of the casino affiliates by the operators (i.e. online casino owners).

When the contract between the SA and the Israeli organisations was terminated, the client organisation signed a new contract with O2, who then took over software development tasks on the applications in February 2008. The team was then using the Waterfall methodology to develop these application projects.

In April 2009, a new project was initiated by the Project Manager in Durban to migrate the "casper" application to a new platform. This project was called the "Colin" application. The "casper" application software, initially developed in C# within the DotNet 1.5

platform, was being migrated to the DotNet 3.5 platform to enhance its maintainability and manageability. The "Colin" application project also involved the addition of new functionalities. The new requirements were obtained from the casino operators. The SA team members were in constant contact with five operators located in Europe. The requirements were then passed on to the Indian team members by the Project Manager in SA. The Scrum methodology was chosen to manage this project. A summary of the Colin Application project is provided in Table 4.8.

Project Name	"Colin" Application Project
Project Purpose	The "Colin" application project primarily involved the migration of the "casper" application, initially developed in C# within the DotNet 1.5 platform, to the DotNet 3.5 platform. The C2 team had to ensure that all the functionalities of the "casper" application had been correctly migrated and that these functionalities worked similarly in the new version. The C2 team also incorporated new functionalities into the application, based on the requirements being put forward by the operators.
Project Initiator	Project Manager / Scrum Master located in Durban, South Africa (Customer)
Project Start Date	April 2009
Methodology employed	Waterfall Approach: February 2008 – January 2009 Scrum: April 2009 – time of case study – For Colin Application Project
Team composition	Project Managers / Scrum Master, Product Owner, Onsite Coordinator, and some Testers located in Durban, South Africa Software Developers, Technical Specialist, Project Manager, and Testers located in Pune, India
Programming Language	C# and DotNet 3.5 platform
Architecture	Model View Controller (MVC)

Table 4-8 – "Colin" Application Project description

4.4.3. The C2 Work Setting

The interviews and observations were undertaken at the office premises in India. On the first day of the case study, the team was located in a smaller office while their new office space was being renovated. The team moved to new offices on the second day.

The smaller office space in the initial offices was not conducive to inter-team communication. The desks were separated by dividers which limited ease of communication. Team members always stood up and looked over the dividers to talk to each other.

The new office, called "agile lab", was more spacious than the previous one and was designed to accommodate several agile teams. The team was seated in one section of the room across a long oval table. The desks were not separated by dividers and hence, communication was not hindered. There were white boards across the walls and provisions had been made for agile artifacts to be displayed. An XP team was also based in the same office space at the time of the study. The Project Manager's office cubicle was located in the same office space, but rather far from the Scrum team.

The meeting room was smaller than the central office space, but was still located in the Agile Lab. The only furniture in the room was a rectangular table surrounded by seven chairs. There was a white board on one side of the wall, on which tasks and their status were jotted down. A phone was also available for the daily Scrum meeting phone calls. The phone was not suited for conference calls, and the team members simply placed it on speaker phone for everyone to hear the conversations.

At some point during data collection, the Project Manager decided to install a desktop computer with a webcam and Skype options to facilitate the daily Scrum meeting phone calls. However, the desktop computer was never utilised. The team members claimed the non-utilisation was due to limited bandwidth.

Video conferences with team members from the customer site were held in a different room located just outside the Agile Lab. The room was equipped with a large flat TV screen, on which stood a video conference camera. A V-shaped table was located in the centre of the room and a conference call phone was placed on the table. The room was equipped with comfortable chairs and top-of-the range telecommunication equipment.

4.5. Data Analysis

Data analysis was guided by the hermeneutics principle (Radnitzky, 1970). Hermeneutics suggests a way for understanding textual data by ascribing meaning to it (Radnitzky, 1970). It is the dialectic between the understanding of the text as a whole and the interpretation of its parts in which descriptions are guided by anticipated explanations (Gadamer, 1976). As mentioned by Taylor (1976), from the hermeneutics perspective, interpretation is an attempt to make sense of the data. Hermeneutics was deemed appropriate as it attempts to make sense of the whole, and the relationship between people, the organisation and information technology. This is useful as in an organisation, people have confused, incomplete, cloudy and contradictory views on many issues. Through hermeneutics, interpretation which consists of "deciphering the hidden meaning in the apparent meaning, in unfolding the levels of meaning implied in the literal meaning" (Ricoeur, 2004, p. xiv) can be achieved. However, concepts were labelled with care as individuals often impart preconceived meanings to terms. This, in turn, might lead readers to deviate from the originally intended meaning ascribed to the concept by the narrator.

Based on what was experienced during the field visits, the researcher also internalised important information which proved useful during analysis. This information might not have been explicitly captured as field notes or during the interview sessions, but was nevertheless internalised and provided a general feel about the teams and why participants behaved in certain ways.

The data analysis was divided into four stages, namely:

- STAGE 1 - Thematic Analysis to identify Scrum process breakdowns during and after Sprint planning and retrospective meetings in C1
- STAGE 2 – Analysis towards understanding the work practices at C1 and the social conditions leading to the Scrum process breakdowns through the lens of Bourdieu's Theory of Practice
- STAGE 3 – Thematic Analysis to identify Scrum process breakdowns during and after sprint planning and retrospective meetings at C2.
- STAGE 4 – Analysis towards understanding the work practices at C2 and the social conditions leading to the Scrum process breakdowns through the lens of Bourdieu's Theory of Practice

The flow of activities relevant to each data analysis phase has been depicted in Appendices 14, 15, 16 and 17. Some of the steps in the data analysis might have occurred in conjunction with other data analysis activities, but were depicted sequentially to simplify the diagrams. In the following sub-sections, the various stages of the analysis and the steps undertaken during these stages are described. The sub-sections provide minimal information on the findings, and are instead geared towards providing clear insights on the analytical processes followed. In-depth descriptions of the findings are later provided in Chapters 5, 6 and 7.

4.5.1. STAGE 1: Analysis to Identify Scrum Process Breakdowns at C1

Stage 1 used thematic analysis to identify the Scrum process breakdowns occurring during and after sprint planning and retrospective meetings. Thematic analysis required a high degree of involvement and interpretation from the researcher because it involved more than just counting words or phrases from the interview transcripts and field notes (Guest, MacQueen, & Namey, 2011).

Due to the degree of interpretation required in thematic analysis, the issue of reliability is often raised when researchers claim to employ this method (Guest et al., 2011). To ensure a systematic approach in identifying the Scrum process breakdowns, and reliability of the results, the study proposed a preliminary definition of what is meant by the term "Scrum process breakdown". This served to ensure that consistent categorical decisions were being made over time (Weber, 1985).

As previously mentioned, a Scrum process breakdown refers to any form of deviation from the ideal Scrum process as specified by the methodology guidelines. In addition, these deviations should lead to some forms of social challenge, conflict or disagreement between team members and/or stakeholders, in order for them to be considered as breakdowns. Appendix 18 describes particular forms of interaction which should occur during an ideal sprint planning and retrospective meeting, and examples of possible deviations which could lead to Scrum process breakdowns. These examples only served as a guideline to the analysis process and the researcher was open to any other forms of deviation appearing from the data. Reliability of the results was also ensured through a detailed description of the analysis process, to demonstrate a link between the data and the results (Polit & Beck, 2004). Data analysis was facilitated by NVivo 8 qualitative analysis tool. All interview transcripts, field notes and meeting recordings were uploaded in NVivo 8. The steps outlined in Appendix 14 are further described below.

The first step in Stage One of the analysis process was the identification of themes around specific instances of Scrum Process breakdowns (see Appendix 14). The interview transcripts were first analysed using information provided in Appendix 18 as a basis to identify the Scrum process breakdowns from the empirical data. Strauss and Corbin (1998) called this process "open coding" as it requires the researcher to be "open to the data" (Patton, 2002, p. 454).

As the coding process proceeded, empirical data from more interview transcripts were analysed. In doing so, the analysis progressed to Step Two, where constant comparison between the accounts of the different respondents was made. For example, when two respondents reported a similar Scrum process breakdown incident, the researcher compared the two accounts to identify the extent of their relationship and similarity. The analytical categorisation process also began at this stage, whereby themes on Scrum process breakdowns which were found to be similar were regrouped into the same category (Bitner, Booms, & Tetreault, 1990). The categories were labelled based on the nature of the similarity between the themes (Bitner et al., 1990). An example of a category along with extracts of some interview statements from C1 matching that category is provided in Appendix 19.

Seemingly divergent accounts were also compared, to identify the extent to which they differed. This process was useful in identifying whether to assign a particular theme to an existing category or whether to create a new category for that theme in order to reflect as many nuances in the data as possible. This further served to strengthen the reliability of the findings by ensuring that the researcher was consistently assigning themes to the right categories (Gremler, 2004).

Scrum process breakdowns reported in the interviews were also compared to what had been noted in the field notes and the minutes of previous meetings, as well as to meeting recordings for further validation. This process was invaluable as it allowed the researcher to obtain a preliminary understanding of the social conditions under which these Scrum process breakdowns occurred. Any Scrum process breakdowns which were not corroborated across the various sources of empirical data were discarded (Step Three). For a theme to be valid, it had to be corroborated in at least two sources of evidence. As this rule was systematically applied while deciding on how to discard an incident, systematisation was achieved (Gremler, 2004).

As the analysis proceeded, the relevance of certain categories emerged. A category was deemed relevant if it encompassed (1) several themes grounded in data, or (2) few themes, but each theme having several extracts of empirical data to support its relevance. Categories which did not comply with these rules were discarded in order to achieve systematisation (Gremler, 2004). In Step Four, a code book on the Scrum process breakdowns was created and updated for later use during the analysis of the data from the second case study (see Appendix 20). The Four steps were iterative until no more empirical data was available for analysis. A summary of the Scrum process breakdown categories and the concepts belonging to these categories are provided in Appendix 21.

4.5.2. STAGE 2: Analysis to Understand the Work Practices and Scrum Process Breakdowns at C1

The second stage of the analysis was dedicated to the understanding of the work practices at C1 and the social conditions leading to the Scrum process breakdowns through the lens of Bourdieu's Theory of Practice. Preliminary definitions of each of the concepts were proposed to facilitate their identification in the data and was summarised in a code book (see Appendix 22). Even though a code book was employed, the definition of the concepts was kept as loose as possible to allow all the richness of the interpretation to emerge from the data.

Step One of the coding process began with the identification of specific themes around the main concepts provided in the code book (Braun & Clarke, 2006). For example, when looking for themes around the concept of *Symbolic Capital*, the researcher constantly reverted back to the definition of that concept in the code book. Given this definition, evidence of various forms of *Symbolic Capital* was sought in the empirical data. Themes around the concept were extracted from the empirical data, iteratively and reflexively compared and contrasted to each other and categorised based on the degree of similarity between them. The categories were grouped according to the specific Theory of Practice concept which they represented. In essence, the same principles described in Stage One, to identify and categorise themes and ensure reliability and validity of the findings, were followed during this stage of the data analysis.

The concepts from the Theory of Practice are so interconnected and mutually reinforcing of each other, that they could not be identified in isolation, as shown in Appendix 15. For

example, in order to have a correct understanding of the positions in the field, the habitus which these positions internalised had to be considered. Furthermore, habitus could not be identified without investigating the practices of the GASD participants. Practices themselves could only be considered by relating the social conditions in which the habitus that generated them was constituted, to the social conditions in which it was implemented. The various forms of capital which an agent possessed also served to identify the position which a GASD participant might hold in the field and by investigating this relationship more closely, it became possible to identify the various strategies which these participants employed to defend their positions. While investigating the themes around the concepts of field, position in the field, habitus, practices and capital, the researcher was constantly reflecting on the forms of misrecognition, double-meaning strategies and symbolic violence occurring in the data. Themes around these concepts were also developed and categorised.

This stage of the analysis required deep reflection and several iterations. Themes and categories were consistently compared to each other as their relevance and validity could only be ascertained if they could all be related to each other harmoniously. At this stage, the researcher was still unable to visualise whether this form of harmony had been achieved, even though all the data sources had been analysed. The circuits of reproduction of the work practices were drawn to ascertain the validity of the relationship between the various categories belonging to each concept. The researcher reflected on these visual representations of the work practices and refined them, by revisiting the themes and their categorisation if required. An example of such a circuit of reproduction is provided in Figure 4.4. When the work practices' circuit of reproduction could no longer be refined, the researcher proceeded to the second step of Stage Two.

Step Two was the identification of the social conditions under which the Scrum process breakdowns occurred. By immersing herself in the identification of the Theory of practice concepts from the empirical data, the researcher had already acquired insights on possible explanations for the Scrum process breakdowns. It was thus not a top-down approach from Step One to Step Two as depicted in Appendix 15. A summary of the meta-categories, categories and concepts identified while analysing the circuits of reproduction of the practices are summarised in Appendix 23.

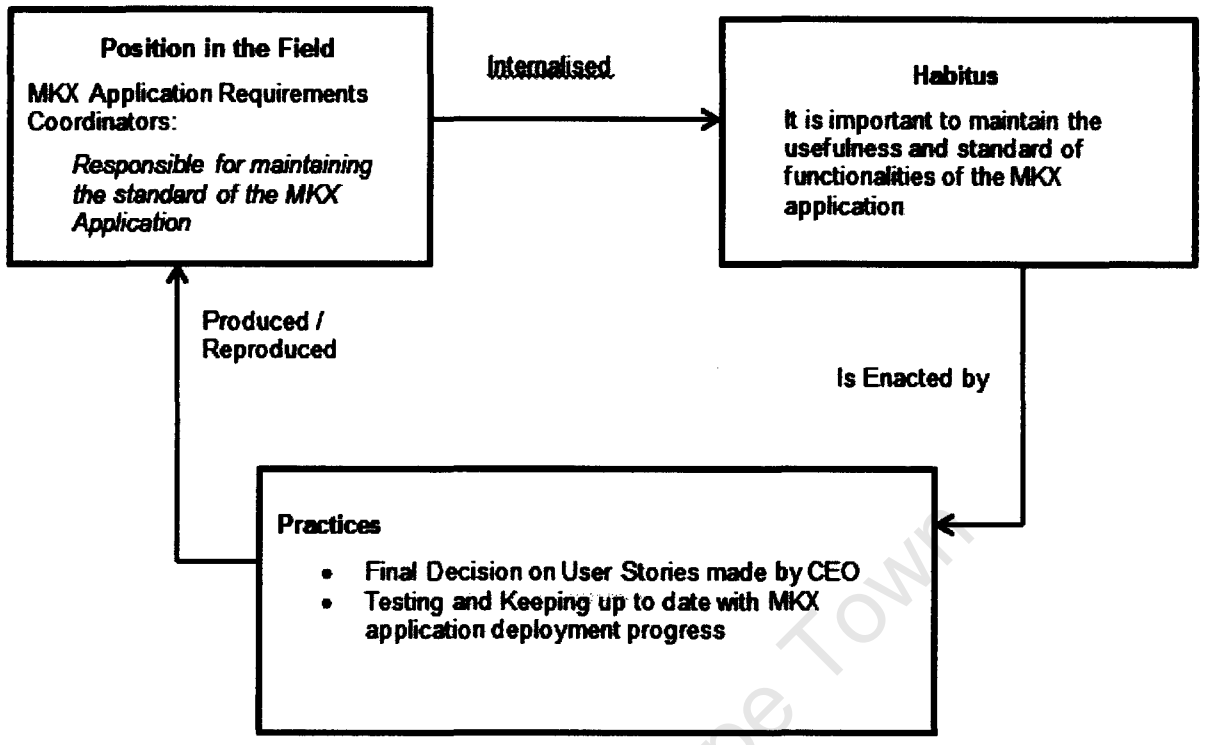


Figure 4-4 - Example of a circuit of reproduction of some work practices in C1

Step One was systematically reviewed to investigate:

(1) The symbolic, economic and cultural capital of the GASD participants experiencing or causing the Scrum process breakdowns in their various overlapping, joint and nested fields.

(2) Whether double-meaning strategies were being employed when these Scrum process breakdowns were occurring and why

(3) Whether symbolic violence occurred when these Scrum process breakdowns were experienced

(4) Whether misrecognition occurred when these Scrum process breakdowns were experienced

(5) The habitus of the GASD participants experiencing or causing the Scrum process breakdowns in their respective overlapping, joint and nested fields.

Given that all the concepts had already been identified in Step One while the circuits of reproduction were being unravelled, Step Two only focused on an interpretation of the empirical findings from Step One. It required that the researcher be highly reflective and attentive to the circumstances under which the Scrum process breakdowns occurred as

seen in the empirical data. Appendix 24 summarises the Scrum process breakdowns during and after the sprint planning and retrospective meetings and the social conditions under which they were seen to occur in Case 1. These further served to assist in the analysis of the empirical data in Case 2. Stage Three of the analysis process began once that this was completed.

4.5.3. STAGE 3: Analysis to Identify Scrum Process Breakdowns at C2

Stage Three of the analysis focused on identifying Scrum Process breakdowns at C2. The steps undertaken during Stage Three are illustrated in Appendix 16. This was achieved by corroborating in C2 the Scrum process breakdown instances identified in C1 (Step 1). This served to assess the validity and authenticity of these Scrum process breakdowns identified in C1 (Patton, 2002). The code book created at the end of Stage One of the data analysis was employed for this purpose. This code book contained a high-level definition of the various categories of breakdown which were identified in C1.

In particular, each source of empirical data was analysed to identify themes matching the categories provided in the code book. For example, when investigating the category *Decisions on Scrum Process Update not made by the Development Team*, the researcher scrutinised the empirical data for evidence of incidents matching this category's definition. New themes matching this category were extracted and iteratively and reflexively compared and contrasted to each other, to form sub-categories of the *Decisions on Scrum Process Update not made by the Development Team* category. The same principles, previously described to achieve reliability and validity of the categories, were employed.

However, the researcher also ensured that she remained open to the emergence of new themes relative to Scrum process breakdowns relevant to C2. This required some form of thematic analysis similar to the process which has been described in Section 4.5.1.

Step Two particularly involved constant comparison of the themes which emerged across the interview transcripts, field notes and observation recordings. Even though in Appendix 16 this step was depicted as a separate process to the identification of themes on Scrum process breakdowns, it was undertaken in conjunction with Step 1 as the analysis progressed. As new themes emerged, they were compared and contrasted to existing ones to achieve internal homogeneity (the extent to which a theme belonged to a category) and external heterogeneity (the extent to which the differences between

categories are clear) (Patton, 2002). In Step Three, any theme identified from the empirical data from C2 which could not be corroborated across the various data sources (using the same criteria described in Section 4.5.1) was discarded.

In Step Three, any Scrum process breakdown category identified in C1 which could not be identified in C2, or which was identified in C2 but not in C1, was carefully recorded for later investigation during Stage Four of the data analysis process. Again, step Four was depicted as a separate process for clarity purposes, but happened in conjunction with all the other data analysis steps. Stage Three progressed until no more empirical data sources were available, after which Stage Four began. A summary of the Scrum process breakdown categories and the concepts belonging to these categories are provided in Appendix 25. A detailed comparison of the Scrum process breakdowns identified in C1 and C2 will be provided in Chapter Seven.

4.5.4. STAGE 4: Analysis to Understand the Work Practices and Scrum Process Breakdowns at C2

Stage Four of the data analysis involved understanding the work practices at C2 and identifying the social conditions under which the Scrum process breakdowns identified in Stage Three occurred. This stage of the data analysis has been depicted in Appendix 17. In Step One, the analysis focused on the identification of various concepts from the Theory of Practice in the empirical data, as a means to understand the logic of the work practices in the various overlapping, joint and nested fields. The same analytical processes employed in the analysis of the empirical data from C1 and described in Section 4.5.2 were used. A summary of the categories and concepts identified while determining the circuit of reproduction of the work practices at C2 have been provided in Appendix 26.

Step One provided the researcher with a deep understanding of the various overlapping, joint and nested fields, and the circuit of reproduction of the work practices within these fields. The researcher could then reflect on how the social conditions inherent in these fields could lead to Scrum process breakdowns (Step Two). At the end of Stage Two (see Section 4.5.2), a tables summarising the social conditions leading to Scrum process breakdowns in C1 were presented (Appendix 24). These tables were used as a means to guide this particular step in the data analysis. While reflecting on the Scrum process breakdowns identified in C2, the researcher sought to identify whether similar social conditions to C1 could be perceived in C2. A similar process to what was identified in

Section 4.5.2 was employed. Any social condition which was seen to lead to a Scrum process breakdown in C2 was compared to those identified in C1 for the same breakdown to enhance the reliability and validity of the results. The researcher systematically investigated each Scrum process breakdown for C2. When a social condition leading to a particular Scrum process breakdown was corroborated in both C1 and C2, this condition was confirmed and was later used in the formulation of theoretical propositions around the Scrum process breakdown phenomenon.

In Appendix 17, it can be seen that Step Three involved examining the deviant cases (Patton, 2002) whereby the researcher sought to investigate why:

(1) Social conditions which were seen to lead to Scrum process breakdowns in C2 were not found in C1
(2) Social conditions which were seen to lead to Scrum process breakdowns in C1 were not found in C2
(3) Scrum process breakdowns were identified in C1 but not in C2
(4) Scrum process breakdowns were identified in C2 but not in C1

This was not a mechanical process but instead required a high degree of reflection and creativity on the part of the researcher (Patton, 2002). After having examined all the deviant cases, some form of coherence could be perceived in the data, following which the theoretical propositions were formulated. A decision table was used to facilitate the formulation of the theoretical propositions, as shown in Appendix 27.

This section has only provided an account of the analytical processes employed to make sense of the empirical data gathered throughout the case studies. Chapters Five and Six will detail the various overlapping, joint and nested field fields and their corresponding work practices as derived from this analysis process. The Scrum process breakdowns and the social conditions leading to them are described in Chapter Seven.

4.6. Access, Privacy, Confidentiality and Ethics

The study was authorised by the University of Cape Town Ethics Committee. The committee decision was based on an application form, summary of study and consent letters. Consequently, the researcher took every possible measure to ensure that the information gathered for the purpose of this study remained strictly for the purpose of this study alone. The participants, their companies and the information and opinions expressed by them, remained completely anonymous. Furthermore, an interview consent form was presented to all participants of this study for them to read, understand, agree to and sign (See Appendix 28).

4.7. Chapter Summary

A summary of the proposed research design is given in Table 4.9.

Research Context	GASD teams using the Scrum methodology
Research Paradigm	Positivist, Qualitative, Empirical
Research Method	Multiple Case Studies
Data Collection Method	- Semi-structured Interviews - Direct Observation Documentation - Field Notes
Data Analysis	Thematic Analysis
Theoretical Framework	Theory of Practice

Table 4-9 - Summary of Research

5. EMPIRICAL OBSERVATIONS FOR DRIFT CASE (CASE 1)

The first case study (C1) is in line with Bonoma (1985)'s "drift" stage of case study design. This chapter details the findings arising from the analysis of the data, and focuses particularly on the work practices at play within C1. The work practices of C2, the description of the Scrum process breakdowns and the social conditions leading to them are provided in Chapters Six and Seven respectively.

The chapter is structured in six sections. Section 5.1 presents an overview of the various overlapping, joint and nested (sub-fields) fields identified for C1 as well as the relationships between them. The overlapping, joint and nested fields are described in the Sections 5.2 to 5.6, paying attention to the positions, habitus, work practices, and the forms of cultural, symbolic and economic capital relevant to each of them.

Individuals in a field occupy social positions and these positions are structured into a system which constitutes the field (Jenkins, 2002). Again, a field is internally structured according to power relations. In particular, given their degree of access to various forms of capital, positions stand in relationships of domination, subordination, or equivalence to each other (Jenkins, 2002). The nature of positions is inherent in the forms of capital found relevant in a field. In addition, the very existence of a field presupposes that participants in that field believe in the legitimacy and validity of the capital at stake in the field (Jenkins, 2002).

To conceptualise the various fields relevant to the cases, the study specifically sought to identify the various positions constituting these fields. The power relations between the positions were also investigated, given the amount of capital (cultural, symbolic and cultural) relevant to the field and owned by the individuals in these positions.

However, given the small size of the team being investigated, the number of individuals belonging to a particular field was small. To keep the focus on the group as opposed to the individual and report on the collective habitus and work practices, the researcher sought not to drill down to too low a level while identifying the positions relevant to these fields. For example, the *Software Developers* nested field (see Section 5.3) was reported as being constituted of the *Technical Expert's* position (with five individuals). It would have been possible to drill down to a lower level of analysis to identify more specific positions in that field (e.g. *PHP Experts, Front-End Technical Experts* and *Back-End*

Technical Experts) as well as the habitus and work practices relevant to these positions. But doing so would imply identifying individual habitus and work practices, given the small sample size. The only exceptions were when the *Scrum Implementer and Driver (*in the *C1 GASD Team* field), *Management (*in the *C2 GASD Team* field), and *Onsite Coordinator (*in the *Sanbi (C2 Organisation)* field) positions were discussed. Given the key role that these position plays in their fields, they had to be discussed separately even though they were composed of only one member.

However, to allow for a richer interpretation, the discussion around the amount of capital owned by each participant was drilled down to the individual team members. This allowed for a better understanding of the participants' position of power (dominance, equivalence, and subordination) in relation to each other. Understanding the various positions of power was important to make sense of the forms of double-meaning strategies, symbolic violence, or misrecognition at play within the fields. It is acknowledged that for a bigger team, the power relationship between specific positions would have emerged as opposed to between individuals. The same analytical logic was employed for both C1 and C2.

5.1. Overview of Fields Identified for C1

The following five key fields were identified for C1: *C1 GASD Team, Software Developers, Cape Town Team, Sao Paulo Team,* and *Brazilian PHP Software Developers*. Since the team members from C1 engaged in work practices within the GASD context and formed part of a global organisation, four other fields were also identified, which influenced the key fields characterising C1: *HIM Group, GSD Community, Agile Community* and *Scrum Community.* As can be seen in Figure 5.1, the different types of lines were used to distinguish between the various fields boundaries.

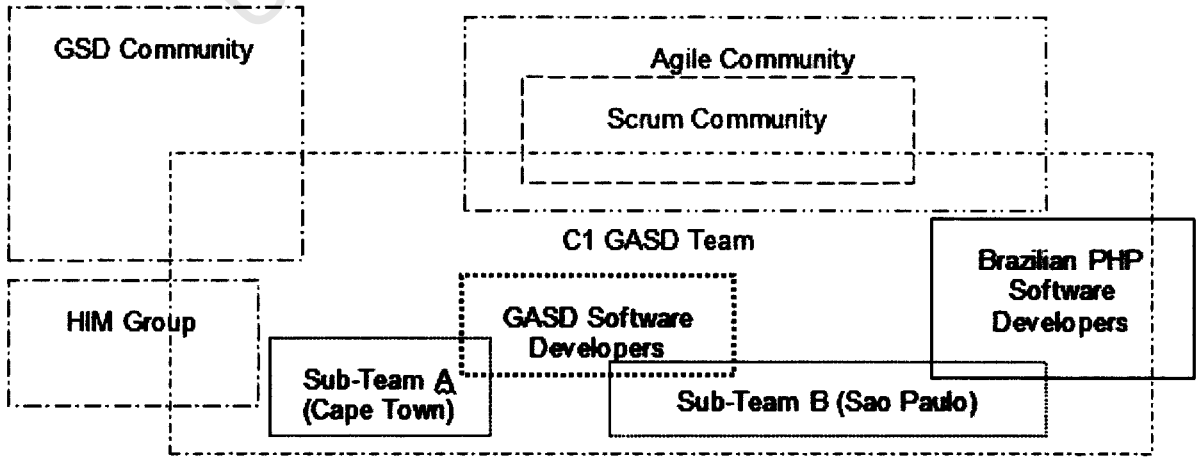


Figure 5-1- Overview of C1 Fields: Overlapping, Joint and Nested Fields

Figure 5.1 illustrates the fields of influence for C1 and the relationships between them. As can be seen in the diagram, team members at C1 belonged to a joint field *C1 GASD Team.* In addition, in line with the two geographic locations, they were also found to belong to two nested fields: *Cape Town Team* and *Sao Paulo Team.* Some software developers from *Sao Paulo team* also belonged to the PHP community, and were thus found to be part of a separate nested field: *Brazilian PHP Software Developers*. Team members who were software developers also formed part of a smaller nested field: *Software Developers.* The *C1 GASD Team* was also influenced by the *HIM Group* field, which encompassed employees of the overall group of companies belonging to the HIM group. In addition, the *C1 GASD Team* field was influenced by the *GSD Community,* the *Scrum Community*, and the *Agile Community* fields. The *Scrum community* field was a nested field of the *Agile Community* field. Each field had specific sets of capital (cultural, symbolic and economic) operating in them.

The following sections describe the logic operating within the five main fields. The sections also detail the different forms of capital (cultural, symbolic and economic) operating in the fields. The *HIM Group, GSD Community, Scrum Community* and *Agile Community* fields have not been described in order to keep the focus on the C1 case.

5.2. The C1 GASD Team Field

The *C1 GASD Team* field is a joint field which emerged when the project was initiated and which is characterised by the following positions: *Scrum Implementer and Driver,* the *MKX Application Requirements Coordinator* and *Software Developers*. Sections 5.2.1 and 5.2.2 describe the habitus and work practices for the positions of *Scrum Implementer and Driver* and *MKX Application Requirements Coordinator*. The habitus and work practices of the *Software Developers* are described in section 5.3 since team members holding this position also belonged to a separate nested field (sub-field) with specific forms of capital. These have thus been described separately in order to avoid repetition.

5.2.1. Scrum Implementer and Driver's Work Practices

To shed light on the work practices of the *Scum Implementer and driver*, each element in Figure 5.2 is described.

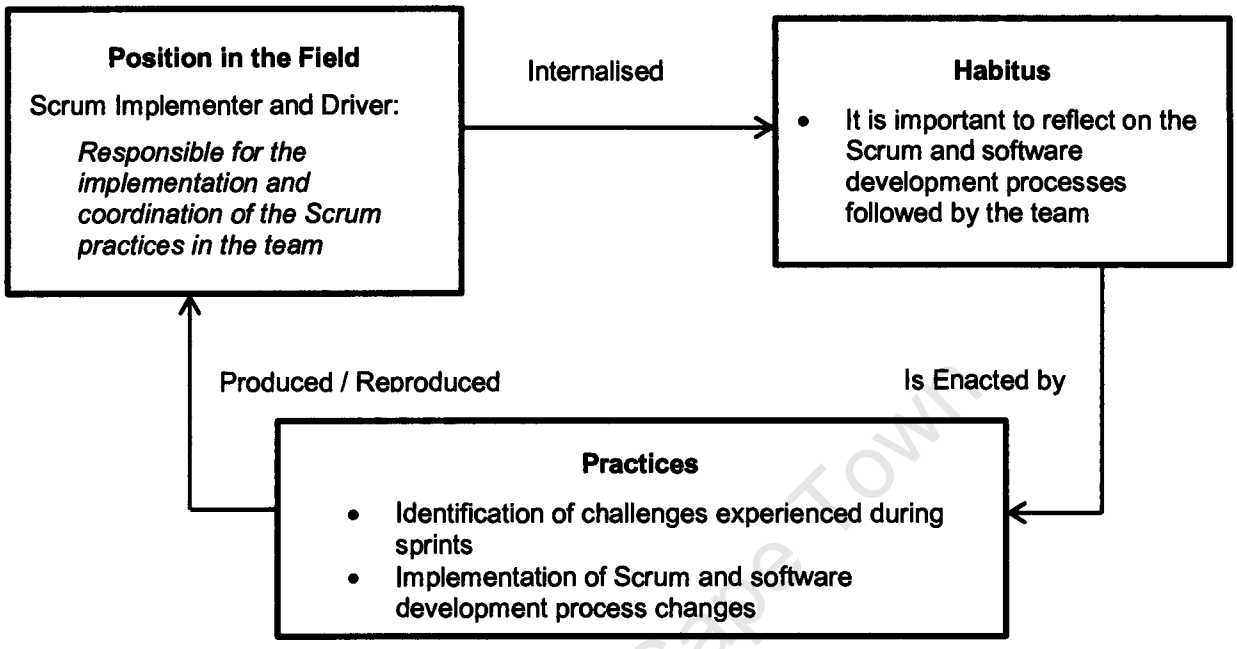


Figure 5-2 - Scrum Implementer and Driver's Work Practices

The *Scrum Implementer and Driver* position was highly relevant but only encompassed one agent namely C1.R1. When the MKX project was first initiated, the team was following the Waterfall approach to software development. C1.R1 was the first to suggest the implementation of Scrum within the C1 GASD team. C1.R1 perceived Scrum as a new and innovative software development trend, appropriate to the nature of the software development tasks that the team engaged in (high rate of requirements change with a need for fast delivery). Consequently, in spite of the lack of experience in Scrum of the whole team (including himself), he *"pushed quite hard"* (C1.R5) for the adoption of that methodology. Initially, the Scrum methodology was implemented *"without any structure or anything"* (C1.R1). Eventually, C1.R1 attended a Scrum Master certification course, and applied the knowledge acquired during that course within the team: *"Then I went on a course, came back and put a little bit more structure to it and that's what we're doing now"* (C1.R1).

Agent	Degree of Domain Application Knowledge	Scrum Certification	Years of Experience in the GASD Team
C1.R1	Limited domain application knowledge as not involved in requirements acquisition and formulation	Yes (Scrum Master)	2 years
C1.R2	Limited domain application knowledge as only recently joined the team	Yes (Product Owner)	3 months
C1.R3	Limited domain application knowledge as not involved in requirements acquisition and formulation. Involved in Scrum Master role	None. Only 1 day of Scrum Master Training	1 year 6 months
C1.R4	Limited domain application knowledge as not involved in requirements acquisition and formulation (High technical knowledge)	None. Attended a 2 day conference on agile software development	2.5 months
C1.R5	High degree of application domain knowledge, as founded the project and is extensively involved in driving the project vision	None	3 years
C1.R6	Limited domain application knowledge as not involved in requirements acquisition and formulation. (Responsible for PHP back-end software development tasks on the MKX application)	None	1 year 6 months
C1.R7	Limited domain application knowledge as not involved in requirements acquisition and formulation. (Expert in front-end software development for the MKX application. However, is now also involved in some back-end software development tasks)	None, only attended 1 conference on Scrum	1 year 8 months
C1.R8	Limited domain application knowledge as not involved in requirements acquisition and formulation (PHP Software development, and management of the software architecture of the MKX application)	None, only attended 1 conference on Scrum	1 year 9 months

Table 5-1 - Cultural Capital for C1 GASD Team Field

For the past two years, he strategized to retain his position as *Scrum Implementer and Driver* in the team (as will be described in Section 5.2.2) and, with the full support of the entire team, drove the evolution of the Scrum and the software development process followed at C1. The *Scrum Implementer and Driver* was in a dominant position in the field and retained that dominant position since he possessed a high amount of relevant cultural and symbolic capital. As can be seen in Table 5.1, he acquired a high amount of cultural capital because of the "Certified Scrum Master" title (Scrum Master) which he obtained after attending the Scrum Master Certification course. He also possessed a high amount of cultural capital pertaining to his two years of experience in the team at the time of the study. He was the only agent possessing some form of symbolic capital in the *C1 GASD Team* field (hence symbolic capital was not represented in a table). The

symbolic capital pertained to the extent to which an agent participated in the implementation of Scrum in the team. He always ensured that he was the one driving the changes made to the Scrum and software development processes.

The economic capital relevant to the *C1 GASD Team* field pertained to an agent's degree of control of the MKX Application. The degree of economic capital owned by the agents in the *C1 GASD Team* field was not represented in a table, because C1.R5 was the only agent who owned this form of capital in the field. The economic capital for the C1 GASD Team field will be described in section 5.2.2.1.

5.2.1.2. Habitus and Practices of the Scrum Implementer and Driver

The habitus of the *Scrum Implementer and Driver* in the *C1 GASD Team* field was:

- *It is important to reflect on the Scrum and software development processes followed by the team*

The habitus was probably transposed from the *Scrum Community* and the *Agile Community* fields. In particular, agents joining *Scrum* Community and the *Agile Community* fields could embraced the habitus whereby it is recognized that any agile methodology, and particularly Scrum, requires that, at regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behaviour accordingly (Koch, 2005). The practices that the *Scrum Implementer and driver* was predisposed to engage in, given this habitus, were the *identification of challenges experienced during Sprints* and the *implementation of Scrum process Changes.* The practices are further discussed below.

- ***Identification of the challenges experienced during sprints***

Agents in the C1 GASD team were given the opportunity to express the challenges and issues they experienced during a particular sprint at a retrospective meeting. These meetings were held at the end of each sprint, as prescribed by the Scrum methodology. Prior to a retrospective meeting, team members would email a list of concerns (challenges and issues) to the *Scrum Implementer and Driver* (C1.R1), who would then compile an agenda for the meeting. Through this arrangement, the *Scrum Implementer and Driver* coordinated the whole process and remained in control by always being aware of the concerns of the rest of the team. He was thus able to reflect and identify possible solutions to these challenges prior to the meeting and would present them to the team during the meeting.

During the retrospective meeting, team members discussed their concerns. They also pointed out what went well during the previous sprint, as a means of ensuring that the practices which they found useful were retained in the upcoming sprints. The feedback obtained during the meetings were consolidated into one list by the *Scrum Implementer and Driver*: *"Usually during the retrospective we tend to look back and see what went well, what wasn't good, take that feedback and pass it on to C1.R1" (C1.R7)*. The retrospective meetings were open to anyone who wished to attend.

Some team members did not always wait for the retrospective meeting to express their grievances. This was particularly the case for team members with high amount of cultural capital pertaining to number of *years of experience in the GASD team*, and the forms of capital relevant to the degree of expertise on the MKX application or a particular type of programming language. The last two forms of capital were specifically relevant to other nested fields (*Software Developers, Cape Town Team*, and *Sao Paulo Team* as will be discussed later), but were nevertheless used to empower the team members to express their views:

> *"Even if they're not happy about the way one has dealt with something, we've seen it before that even during sprints, they raise the issue. They don't wait till the retrospective. They will say: 'I will do this, but you know I wasn't happy with this and we shouldn't do this'" (C1.R8).*

- ***Implementation of Scrum and software development process changes***

The *Scrum Implementer and Driver* ensured that changes to the Scrum and software development processes, followed by the team, were implemented in various ways throughout the course of the project. For example, at the time of the case study, it had become an established practice within the C1 GASD team that a task would no longer be estimated in days but in hours. This decision, even though not unanimously approved by all the team members, was made by the *Scrum Implementer and Driver*, and was a strategic move allowing him to have a better perspective on the team's performance: *"Actually C1.R1 as the chief operations officer feels like he needs to have a look at the hours"* (C1.R6). The team members also reported that during the first few sprints when Scrum was first implemented, the daily Scrum meetings, recommended by the methodology, were not always conducted. Given his high amount of symbolic capital as Scrum implementer in the team and his dominant position in the field, C1.R1 decided to

enforce these meetings: *"One of the things that we need to stick to is to have the daily standups (don't skip it) have the sprint planning. That's when you get the feedback. Otherwise you wait for too long" (C1.R1).*

Other changes implemented by the *Scrum Implementer and Driver* were the adoption of the Scrum tool VersionOne to manage the task-board and product backlog, as well as the adjustment of the sprint length from two weeks to four weeks. The reasons given by C1.R1 for implementing these decisions related to the globally distributed setting within which the team members engaged in and the need to bridge the distance between them: *"We just moved from two weeks to four weeks as we realized that especially with distributed, two weeks was just a little bit too short" (C1.R1).*

In all instances, it appeared as though the practices of the *Scrum Implementer and Driver* were triggered from a strategic intent to better manage the team and increase the chances of yielding project success. However, these practices served to reproduce the structure of the field that produced them, and reinforced the *Scrum Implementer and Driver*'s position in the field as implementer and coordinator of the Scrum work practices. By implementing various changes to the Scrum process, the *Scrum Implementer and Driver* remained in control, reasserted his influence and dominant position in the C1 GASD field, and also acquired more capital within that field. All team members were under the impression that the Scrum process implemented in the team was always updated in a democratic manner given the requirements of the team members: *"It's a group decision. We put forth recommendations..." (C1.R6).* However, an analysis of the situation showed that this was not always the case. Each particular form of process update was a strategy employed by the *Scrum Implementer and Driver* to maintain his dominant position in the field and thus reproduce the field's structure.

5.2.2. MKX Application Requirements Coordinator's Work Practices

To describe the circuit of reproduction of the *MKX Application Requirements Coordinator's* work practices, each element Figure 5.3 is described.

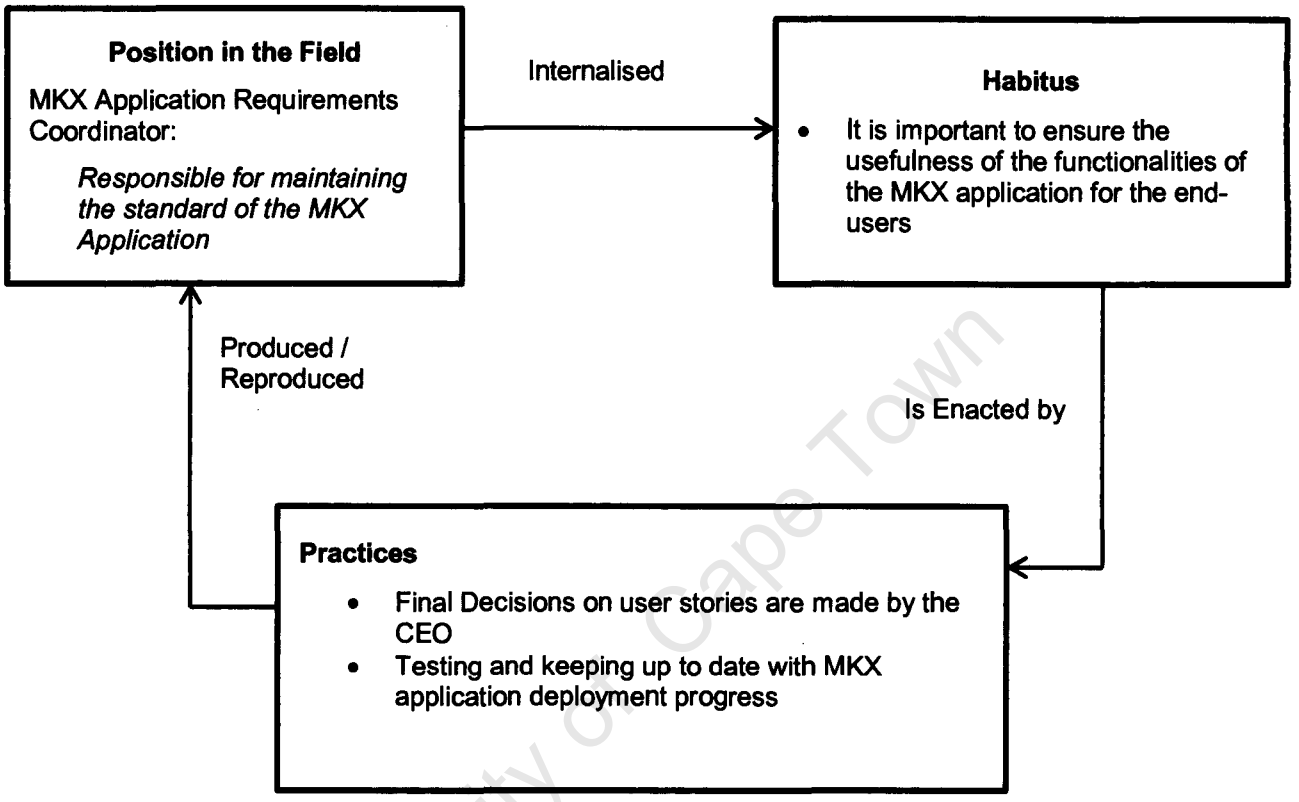


Figure 5-3 - MKX Application Requirements Coordinators' Work Practices

5.2.2.1. MKX Application Requirements Coordinator Position in the Field

The *MKX Application Requirements Coordinator* position was held by C1.R2 and C1.R5. Agents in this position were perceived as the ones responsible for maintaining the quality and usefulness standard of the MKX application in the *C1 GASD Team* field. C1.R2 was the Product Owner and was in a subordinate position. On the other hand, C1.R5 was the CEO and customer for the project and was in a dominant position. He also defined himself as the strategic driver of the MKX application:

> *"In terms of the guiding vision, I think that it's my responsibility to take this project, this particular product that you're referring to (MKX), and strategically position it to make sure that it meets the business needs of the company and also of this business unit"* (C1.R5).

C1.R5's dominant position stemmed from the following cultural capital, of which he possessed a high amount: *Degree of Domain Application Knowledge,* and *Years of Experience in the GASD Team*. C1.R5 also possessed a high amount of symbolic capital related to his *Job Title;* he was CEO. In addition, he was the only agent possessing a high amount of economic capital within the *C1 GASD Team* field. In particular, he was the initial founder and Product Owner of the MKX application. Funding money was not obtained from his salary but from managerial funds of which he had substantial control. Given that he had been the initial Product Owner of the MKX application for the past two years, he retained a high degree of interest in the product.

At the time of the study, C1.R2 had been a member of the C1 GASD team for only three months. He was in a subordinate position in the field given his low level of cultural capital pertaining to his *Degree of Domain Application Knowledge* and his *Years of Experience in the GASD Team*. He also possessed limited economic capital as he had limited control of the MKX application.

C1.R5 employed the MKX application as a tool to maintain his dominant position not only in the *C1 GASD Team* field, but also in the *HIM Group* field. In particular, C1.R5 claimed that his involvement in the MKX project was purely at a high level where he was driving the vision of the product. However, in practice he was much more invested in the team and prescribed both specific requirements and product priorities. This can be perceived as a form of double-meaning strategy whereby he claimed to be working in the interest of the C1 GASD team but was in fact pursuing his own agenda to maintain his position in the bigger HIM group as innovation leader.

5.2.2.2. Habitus and Practices of the MKX Application Requirements Coordinator

The habitus of the *MKX Application Coordinator* in the *C1 GASD Team* field was:

- *It is important to ensure the usefulness of the functionalities of the MKX application for the end-users*

The habitus internalized by agents in this position was the belief that it is important to ensure the usefulness of the MKX application's functionalities for the end-users. In particular, agents in that position felt that the MKX application should be of a high standard, and strived to identify useful functionalities for the end-users. In turn, it was commonly perceived that the functionalities developed for the MKX application should contribute towards creating synergy across the different companies in the HIM group and

promote knowledge management: *"We are very much into knowledge management. We've created a knowledge management tool and this is what we are working on. The tool is trying to get everybody from the different companies to talk to each other"* (C1.R1). The practices that agents in the *MKX Application Requirements Coordinator* position were predisposed to engage in were: *Final Decision on User Stories are made by the CEO* and *Testing and Keeping up to date with MKX application deployment progress*. These are further described below.

- ***Final decision on user stories are made by the CEO***

The analysis identified that, in his position of *MKX Application Requirements Coordinator*, C1.R5 was the key agent responsible for the identification of the requirements, drafting the user stories, and determining the priority of these user stories. As previously mentioned, he defended his position in both the HIM Group and the C1 GASD fields by pushing for the implementation of his ideas on the MKX application.

C1.R5 gathered ideas pertaining to the requirements for the MKX application in many ways. For instance, he often travelled to the various sites at which the HIM companies were located, to personally meet key end-users and gather their views about the application: *"I pick up requirements from interactions with other people in the group. So there are things that I pick up from some of the other executives"* (C1.R5). He also claimed that he not only obtained requests directly from users but also from the Product Owner (C1.R2), who was also tasked with *"picking up requirements from users"* (C1.R5). C1.R5 defended his position in the field by always retaining control over the requirements gathering and elicitation process. He applied double-meaning strategies by claiming that he could not allow end-users to freely put forward any requirements to avoid chaos:

> *"There's no way where I could practically facilitate a system where anybody from around the world could actually put down their requirements. We could put up something, saying here's a website go put it down there, it would be chaos. Even though we want this application to be as useful as possible, it's not a free for all. So in that sense, we take strategic responsibility to where it's going and drive it in that way"* (C1.R5).

End users were often presented with functionalities, instead of the requirements which they requested:

"It's something that C1.R5 and I have discussed and it's something that I've also brought in: 'think in terms of the customer more'. I don't think that we actually have that interaction with users enough. It does come through on MKX as well, where people say 'why are we doing this? Why can't we do it like that?'" (C1.R2)

Consequently, C1.R5 had the final say about the requirements which would be implemented in the sprint and their corresponding priorities. When asked his opinion about the manner in which the requirements were gathered for the MKX project, C1.R2 expressed his desire to formalize the requirements gathering process:

"I would like to make it a more structured process as well. So as soon as we get our sprint process stable as well, this is something that I want to look into more. To really connect with people more, find out what's working for them, what's not working" (C1.R2).

In doing so, C1.R2 sought to enhance his position in the structure of the field, and acquire more capital in terms of his current role in the *C1 GASD Team* field. Nevertheless, C1.R2 was still able to discuss and share his opinion about the requirements during a meeting held with C1.R5 prior to the sprint planning meeting where all the other team members participated.

- ***Testing and keeping up to date with MKX application deployment progress***

Given their position as *MKX Application Requirements Coordinators,* C1.R2 and C1.R5 regularly participated in testing activities. In particular, C1.R2 was responsible for integration testing while C1.R5 handled the acceptance testing: *"C1.R2 will send stuff to C1.R5 who will do the acceptance testing and ... [C1.R5] won't do the integration testing. [C1.R2] must do the integration as well"* (C1.R8). The deployment of the new releases was also closely monitored by both C1.R2 and C1.R5. By partaking in these activities, both agents enabled the fields' structure to be reproduced.

5.3. The GASD Software Developers Field

The software developers at C1 belonged to a nested field within the main *C1 GASD Team* joint field which was labelled as: *GASD Software Developers.* The *GASD Software Developers* field was characterised by one position namely: *Technical Experts.* The position, habitus, and work practices the *Technical Experts* are described in section 5.3.1.

5.3.1. Technical Experts' Work Practices

In order to describe the circuit of reproduction of the *Technical Experts'* work practices, each element in Figure 5.4 is described.

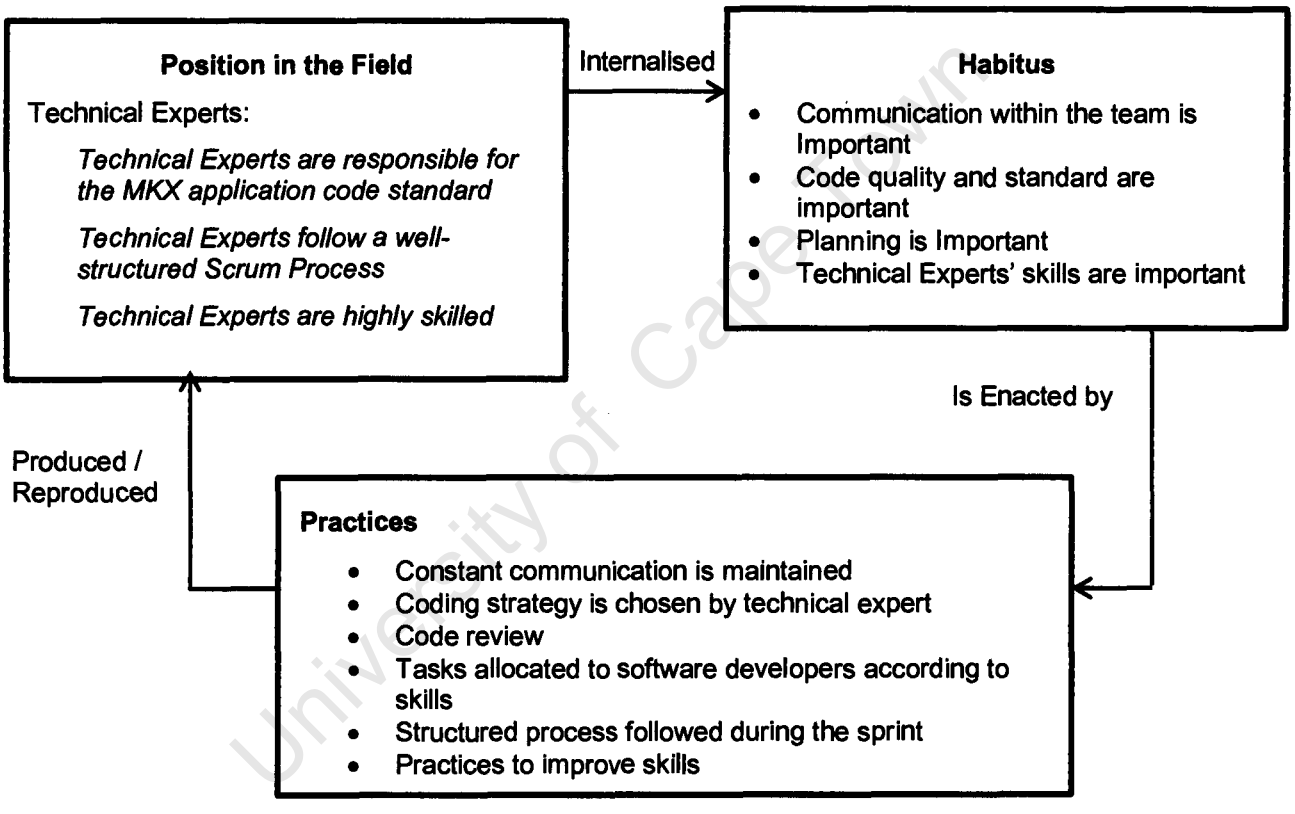


Figure 5-4 – Technical Experts' Work Practices

5.3.1.1. Technical Experts Position in the Field

In the *GASD Software Developers* nested field, *technical experts* were perceived: (1) to be responsible for maintaining the MKX application code standard, (2) to ensure that a well-structured Scrum process was followed, and (3) as being highly skilled.

Agents belonging to the *Technical Experts* position within the *Software Developers* field also belonged to the *C1 GASD Team* field within which they were in positions of

equivalence (C1.R3, C1.R6, and C1.R8) and subordination (C1.R4 and C1.R7). When examined at this level, these agents did not appear to have much influence within the *C1 GASD Team* joint field. However, their degree of influence was higher within the *GASD Software Developers* field as this nested field had its own set of relevant symbolic and cultural capital, of which some of these agents possessed a high amount.

The cultural capital in the field was relative to the agents' *degree of expertise on the software development tasks for the MKX application* while the symbolic capital was relative to the extent to which the agents had a *useful area of expertise* (see Table 5.2 and 5.3 respectively). In particular, C1.R3, C1.R6 and C1.R8 were in dominant positions in the *GASD Software Developers* field. C1.R3 and C1.R8 were both perceived as the MKX application technical experts, and C1.R6 was the backend software development expert. C1.R7 was in an equivalent position and was perceived as the frontend application development expert while C1.R4 is in a subordinate position and was perceived as a novice within this particular team pertaining to his expertise level on the MKX application.

Agent	Degree of Expertise in software development tasks for MKX application
C1.R3	High level of expertise in the technical aspect of the MKX project
C1.R4	Low level of expertise as he only joined the team 2.5 months ago (novice)
C1.R6	High experience in the back-end component of the MKX application
C1.R7	High level of expertise in front-end software development on the MKX application. She has been involved in that role since the beginning of the project. Lower level of expertise on the back-end and PHP development as she is new in the role
C1.R8	Team member with the highest level of technical expertise pertaining to the software development of the MKX application

Table 5-2 - Cultural Capital for the Software Developers sub-field

C1.R3, C1.R6 and C1.R8 were experts in the PHP programming language, while C1.R7 was an expert in Javascript and CSS. C1.R8 was also an expert in the software architecture. C1.R4 had no specific area of expertise.

Agent	Having a useful area of expertise
C1.R3	PHP Software Development
C1.R4	No specific area of expertise except that his job title says that he is the technical architect
C1.R6	PHP software Development
C1.R7	Javascript and CSS (Cascading Style Sheets)
C1.R8	PHP Software Development and technical software architecture

Table 5-3 - Symbolic Capital for the *GASD* Software Developers field

5.3.1.2. Habitus and Practices of the Technical Experts

The habitus internalized by the *technical experts* in the *GASD Software Developers* field related the fact that communication within the team, work quality, planning, and software developers' skills were valued. These are illustrated in Figure 5.4 and are further discussed below along with the corresponding set of practices to which the agents are predisposed to engage in, given the habitus.

- *Communication within the team is important*

Given their positions, whereby they were perceived as the ones responsible for the MKX application code standard, *Technical Experts* internalized the habitus valuing the importance of communication within the team. Such is the case since, from past experience and various socialization processes (Jarwitz, 2008), *Technical Experts* at C1 'embrained' the knowledge that a high level of communication should be maintained to produce software applications of high quality. They learnt the value of communication from their past interactions in the distributed team, as well as from the other software teams in which they were involved. This is in line with the transposabilty and durability of dispositions inherent to habitus whereby it can be relevant to other fields or social contexts than the ones in which it was originally created (Webb et al., 2002). The emergence of the importance of communication can also be explained by the fact that none of the software developers at C1 were novices. They all had prior experience in software development from other organisations where they learnt about the importance of communication within a software development team. When they joined the C1 GASD team, a new field emerged and their individual habitus (pertaining to the importance of communication) was exposed to this specific new field and the same "logic of action" over an extended period of time (Bourdieu, 1990, p. 58). This gave rise to the collective

habitus favoring the importance of communication within the *GASD Software Developers* field. The practices which they were predisposed to engage in given this form of habitus are further described below.

- ***Constant communication is maintained***

Without having been designed to that end, various schemes were generated to maintain constant communication within the *GASD Software Developers* field. In particular, *Technical Experts* at C1 were predisposed to maintain an open communication channel in order to avoid misunderstandings and clashes and to voice any concerns which they encountered during the sprint: *"I think that I spend most of my day chatting to the Brazilians. Also, because C1.R2 is not very technical, I am this in-between person for them to speak to technically and then translate that to C1.R2"* (C1.R3). Because of the geographical distance, various forms of communication tools were also employed to maintain that open communication channel. For instance, emails, video conference applications, instant messengers and Skype phone calls were often used. They thus often identified potential requirements-related issues and misunderstandings earlier-on, and were able to produce complete and adequate functionalities for the MKX application. The structure was thus reproduced as they maintained the high standard of code being written.

In line with the Scrum process recommendations, daily stand-up meetings were also organised throughout the sprint. *Technical experts* explained that this practice was followed as part of the requirements of the Scrum software development methodology, i.e. a Scrum rule. Through this rule, it was forgotten that the technical experts were trying to pursue some personal interests in the field. For instance, while executing the daily stand-up meetings, the structure of the field was reproduced as the agents defended their positions as experts on the MKX application by striving to complete the requirements for the sprint accurately and with minimal number of bugs. The daily stand-up meetings allowed them to constantly monitor their progress, and leveraged their chances of successfully achieving their common goals i.e. completing all the requirements for the sprint:

> *"It's just basically to keep the communication channels open. So I have the daily meetings, make sure that you've got the ways and means technically to do it"* (C1.R3).

- *Code quality and standard are important*

The habitus calling for the importance of code quality and standard was also internalised by the *Technical Experts* whereby they valued the need to maintain high quality and standard for the MKX application software code. Similarly to the communication habitus, they learnt about the importance of code quality and standard from past projects in which they were involved. Technical experts also expressed the need to maintain a good code standard to facilitate the work of the other team members who might also work with the same code base at some point:

> *If everybody is involved with working with the same application, for example today you would be working on a piece of code. And tomorrow somebody else would be working with that piece of code. And because you know that other person quite well, you would rather make sure that it's done properly so that you make sure that it's easier for this person to work with it* [C1.R4].

The practices which they were predisposed to engage in, given this form of habitus, are further described below.

- ***Coding strategy, task allocation, and code review***

The set of practices to which the *Technical Experts* were predisposed to engage in (given their habitus valuing the importance of code quality and standard) related to the coding strategy employed by the team, the task allocation process and code reviews. Through this habitus, the interests of the software developers were harmonized and a common consensus on the meaning of these specific practices was derived.

In particular, because of their high amount of cultural and symbolic capital pertaining to their degree and form of expertise, each team member was able to decide on his or her individual coding strategy in relation to how to complete a particular task or user story. In doing so, they also maintained their positions as experts in the MKX application. Each team member had enough capital to be trusted in making good decisions pertaining to the completion of software development tasks. Programming tasks were also allocated to the technical experts based on the amount of symbolic capital which they possessed (*having a useful area of expertise).* Through this practice, the structure was reproduced as it was ensured that the application produced was of high standard and the respective positions of the agents in the field were maintained.

Code reviews were also held regularly. For the code reviews, team members possessing high amount of symbolic capital (see Table 5.3) would review the codes and algorithms written by their fellow team members and advise them on how to improve the quality of these codes:

> *"because everybody works on the same code base, they would review each other's code and it might not be a formal review, it might just be something that they've seen and they would just IM somebody and say 'Listen, I saw that you're doing this in the code, please do that and that instead cause I've read this article'"* (C1.R4).

Code reviews could be perceived as a form of double-meaning strategy employed by some team members to ascertain their dominant positions as experts in the software development fields. Team members whose codes were being reviewed in turn reported that they were *"motivated" [C1.R5]* when their code was being reviewed, indicating some form of misrecognition or symbolic violence with the agents' accord. They also stated that code reviews enticed them to *"write good codes as other developers will be working on them later" [C1.R5].* This practice, even though not consciously orchestrated to that end by the agents in the field, served to reproduce the *GASD Software Developers* field's structure.

- *Planning is important*

The habitus calling for the importance of planning was internalised by the *Technical Experts* and was transposed from another overlapping field influencing the relevant fields of C1 namely, the *GSD Community* field. Whilst having been part of the GSD community for some time, the *technical experts* had internalized this habitus whose relevance was felt in their daily work: *We plan in the beginning what we're going to release. If we've got stories that go into different sprints, we break everything into manageable chunks* [C1.R6]. The practices which they were predisposed to engage in given this form of habitus are further described below.

- ***Structured process followed during the sprint***

The set of practices, orchestrated by the habitus valuing the importance of planning, related to: the processes followed during the sprint to manage the bug fixing process, to handle task dependencies, and to work on the stories based on their priorities. These practices served to reproduce the structure and the agents' positions in the *GASD Software Developers* field whereby they always followed a well-structured Scrum

process. In particular the need to plan was a commonly accepted belief amongst the technical experts and they consequently made time provision for unexpected bugs which might need to be urgently fixed during the sprint. Various mechanisms were also put in place to manage situations when tasks were seen to be dependent on each other:

> *"If a task is dependent on something that's quite a chunk of work, we'll either stop that task and do the work that's required before that, or if it's a small thing, we'll just do things side by side, giving that dependency to someone else to do and they can do that as their next task* (C1.R6).

In turn, stories of highest priority were worked on first: *"the top priority things are sorted at the top in VersionOne and we work on the top priority stories first"* (C1.R4).

- *Technical experts' skills are important*

The habitus valuing the Technical Experts' skills was internalised by the *Technical Experts* in the *GASD Software Developers* field. They were thus inclined to value the need to maintain their skills level within the team. Agents were expected to create and maintain an expertise niche for themselves and they were given the space and time to do so:

> *Avery person is supposed to position himself or herself as an expert in a specific area* (C1.R5)
> *And we've recently tried to make everybody do everything so that we can all scale up in different areas but there's still people that has the expertise* (C1.R6)

The practices which they were predisposed to engage in given this form of habitus are further described below.

- ***Practices to improve skills***

Technical Experts were predisposed to engage in numerous forms of practices given their habitus valuing the need to improve their skills level. Firstly, in addition to allocating tasks to team members based on their skills, it was observed that tasks were also allocated as a means to promote learning. For example, C1.R7's area of expertise was front-end software development (Javascript and CSS), and most of the time she was allocated these related types of task. At the time of the study, she showed an interest in back-end software development and was thus at times allocated simple tasks using PHP.

Her interest in backend software development could be perceived as a strategy for her to improve her position in the *GASD Software Development* field, and gain more symbolic capital relevant to that field. This strategy was accepted by the entire team as all the members' interests were harmonized around the need to have highly skilled team members, which would in turn enhance the *Technical Experts'* overall position within the wider *C1 GASD team* joint field. C1.R7 was thus slowly eased into the back-end software development arena and was given non-complicated tasks in order not to compromise the quality of the work being produced for the MKX application:

> *"C1.R7 can't work on very complicated things at this stage, because she's only getting into the code. So we're giving her some tasks which are not as complicated, we need to take that into consideration. And also with other team members who have been moved from other projects, they also don't have a lot of exposure in the project, so we have to ease them into it"* (C1.R3).

Knowledge sharing between team members was promoted across the different sites. In particular, *Technical Experts* were either flown from Brazil to South Africa (or vice versa) for a month. During this time together, they focused on sharing their knowledge on their various areas of expertise: *"They sat here for about a month and we actually... the one time we had presentations where we had knowledge sharing with the people, and just work together for a month to get things going"* (C1.R3). Again, this strategy allowed the *Technical Experts* to enhance their position within the field and their ability to better negotiate throughout the Scrum process.

It was also observed that *Technical Experts* in a dominant position in the *GASD Software Developers* field were more likely to be placed in more influential job positions. For example, C1.R3 had a high amount of cultural capital and symbolic capital pertaining to his degree of expertise on software development tasks for the MKX application and the degree to which he had a useful area of expertise. He was thus in a dominant position in the *GASD Software Development* field. Because of his capital, he was able to further enhance his position and be placed in the role of Scrum Master of the team. This strategy was in line with the habitus in the *GASD Software Development* field whereby team members should be highly skilled. The other *Technical Experts* justified this action by stating that they wanted to ensure that placing people in new job roles would allow them to learn: *"what's important to note is that Scrum master is a fairly important job, so if we*

ask somebody to be Scrum master, it is also for a specific purpose to teach that person, to give him the opportunity to learn or to get to a new position (C1.R5).

The *Technical Experts* also enhanced their skills by learning from other members of the wider agile and PHP community. In particular, they attended conferences, followed technical discussion forums online, and read experience reports written by members of the wider agile community:

> *"I find twitter quite an important source of information. Just by following certain people, you really find that things come out. Of course, when you listen to certain people it becomes very apparent how their group strategy changes to adapt to certain things"* (C1.R4).

5.4. The Cape Town Team Field

The team members at C1, based in Cape Town, were seen to belong to a separate nested field labelled as *Cape Town Team.* The *Cape Town Team* field had one position namely: *Cape Town Software Developers.* The position, habitus and work practices the *Cape Town Software Developers* are described in section 5.4.1.

5.4.1. Cape Town Software Developers' Work Practices

In order to describe the circuit of reproduction of the *Cape Town Software Developers'* work practices, each element in Figure 5.5 is described.

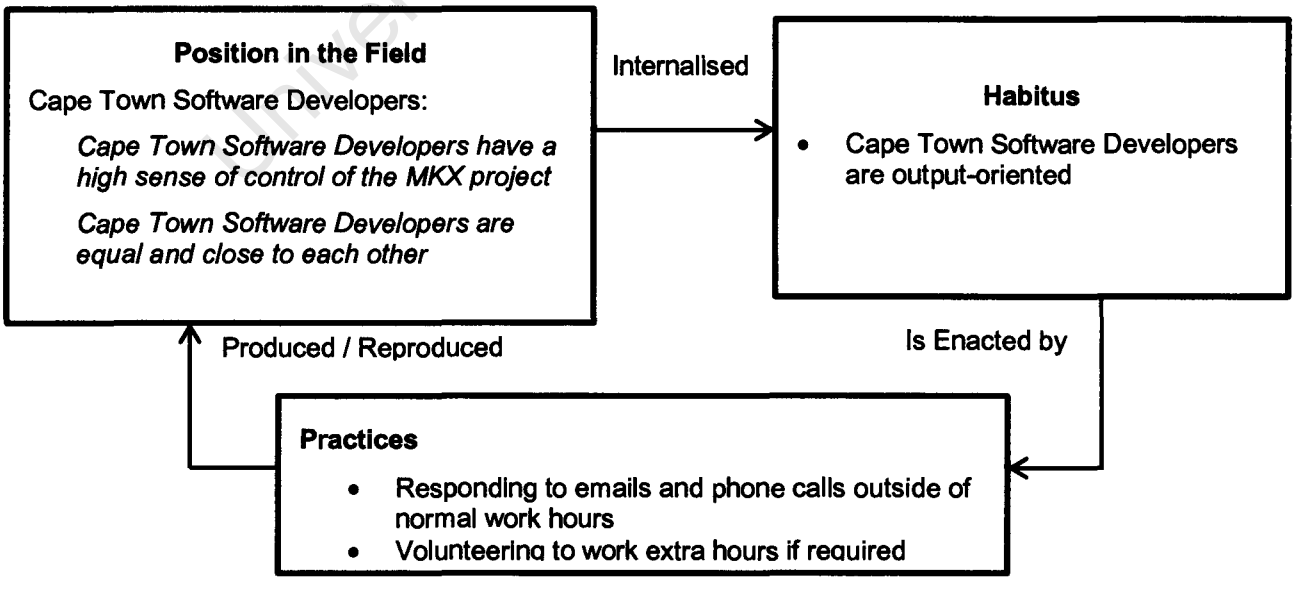


Figure 5-5 – Cape Town Software Developers' Work Practices

5.4.1.1. Cape Town Software Developers' Position in the Field

The position of *Cape Town Software Developers* was held by C1.R2, C1.R3, C1.R4 and C1.R8. The *Cape Town Team* field had its own set of cultural and symbolic capital and the amount of capital possessed by each agent determined whether they were in equivalence, subordination, or dominance position. No significant form of economic capital was found relevant to the *Cape Town Team* field. Tables 5.4 and 5.5 describe the relevant forms of cultural and symbolic capital for the *Cape Town Team* field, respectively.

Agent	Degree of Expertise in software development tasks for MKX application	Degree of Expertise in MKX software application architecture
C1.R2	No technical expertise in software development tasks for MKX. He is the Product Owner and is responsible for conceptualising the user stories	No expertise in the MKX software application architecture
C1.R3	High level of expertise in the technical aspect of the MKX project	Low expertise in application architecture. His main job description is being a software developer and a Scrum Master
C1.R4	Low expertise as he only joined the team 2.5 months ago	His job title is "technical architect" and he thus has experience in general software architecture but not particularly MKX architecture
C1.R8	Team member with the highest level of technical expertise pertaining to the software development of the MKX application	Currently involved in maintaining the software architecture of the MKX application

Table 5-4 - Cultural Capital for the Cape Town Sub Field

The positions held by agents in the *Cape Town Team* nested field were relative to their degree of expertise in software development tasks and application architecture of the MKX application (cultural capital) and years of experience in the MKX application (symbolic capital). The number of years of experience in the MKX application was perceived to be a form of symbolic capital because, more than the knowledge that they

would have acquired while working on that application, it gave the team members some form of prestige and the right to be listened to by others in the team.

C1.R3 and C1.R8 were in a dominant position in that field and were regarded as "experts" on the MKX application. Both C1.R3 and C1.R8 had a high degree of expertise in software development tasks for the MKX application. C1.R8 was responsible for the maintenance of the MKX application's architecture and this role was highly respected within the *Cape Town Team A* field. C1.R3 had low expertise in the MKX application architecture, but his position as Scrum Master contributed to leveraging his overall position of power within that field. Both C1.R3 and C1.R8 had a high degree of symbolic capital within the *Cape Town Team* field as they had both been involved with the MKX project since its inception (*Years of experience in MKX application* symbolic capital).

Given his role as Product Owner, C1.R2 was considered to be the expert in managing the requirements for the MKX application and held an equivalence position of power within the *Sub-Field A (Cape Town)* field. However, in terms of cultural capital, he had no technical expertise in software development tasks and the MKX application architecture. In addition his amount of symbolic capital relevant to that field was also low, as he had only recently joined the team. His equivalence position in the field was only due to his expertise in the requirements management of the MKX application. C1.R4 was the agent with the lowest amount of symbolic and cultural capital within the *Cape Town Team* sub-field and was thus in a position of subordination in comparison to the other agents. His degree of expertise in the MKX application was low as he had only been in the team for three months.

The habitus and practices inherent to the *Cape Town Team* sub-field are further discussed in the next section.

Agent	Years of experience in MKX application
C1.R2	3 months
C1.R3	Has been involved with the MKX application since the beginning of the project
C1.R4	2.5 months
C1.R8	Has been involved with the MKX application since the beginning of the project

Table 5-5 - Symbolic Capital for the Cape Town Team Field

5.4.1.2. Habitus and Practices of the Cape Town Software Developers

The habitus internalized by the *Cape Town Software Developers* in the *Cape Town Team* nested field related to the fact that these agents valued the need to be output-oriented, as illustrated in Figure 5.6. This habitus is further described below.

- *Cape Town Software Developers are output-oriented*

As part of their habitus, the *Cape Town Software Developers* valued the need to be output-oriented. This form of habitus appeared to be imposed on the agents within the *Cape Town Team* field as they felt that it was unconceivable for them not to be highly productive during each sprint and deliver whatever was expected of them by their Product Owner and the CEO:

> *"If you've been given this task and if you say that this will actually take that long to do and you do meet those requirements, it's quite a good feeling, to actually meet the requirements that you've set for myself"* (C1.R8).

The practices which they were predisposed to engage in given this habitus are described below.

- ***Responding to emails outside normal work hours and working extra hours***

In response to the encounter between *output-oriented* habitus and its disposition, as well as the constraints, demands and opportunities within the *Cape Town Team* field, the practice of responding to emails whenever required, even outside normal work hours, emerged. This practice reinforced the structure within which the agents had a sense of control to the MKX project. In addition, the team members felt that in following this practice, they abided by the set of rules in place within the team specifying that the work of their fellow team members in Brazil should not be delayed, as can be seen from that extract:

> ***"Do you respond to their emails at 1am?*** *If I am awake yes. When they need me to be up, they will tell me during the day: 'catch you get me online later to do some stuff'"* (C1.R8).

The invocation of this rule (the practice of responding to emails late at night) is a shortcut to delineate the relation between habitus and the socially constructed situation.

Through this rule, the agents responding to emails late at night forgot that they were in fact pursuing some personal interests which in fact related to maintaining their position of power and affirming their expertise level on the MKX project in the *Sub-Field A (Cape Town)* field as well as the overall *GASD Software Developers* field by advising other team members on pieces of codes. Team members within the *Sub-Field A (Cape Town)* field also volunteered to work extra hours whenever required:

> *"We don't require this from our developers but if need be, for extra hours work, everybody is quite happy to do it. They'll even volunteer to do it. They'll say: 'I haven't managed to do this, it's really important so I'll finish this tonight'"* (C1.R8).

This practice appeared to be motivated by the need to complete the project on time but was also a strategy employed by the team members to acquire more expertise on the MKX application. In particular, by working extra hours, they demonstrated an ability to effectively complete the work assigned to them. This further implied that, in the next sprint, they could potentially be working on more complex tasks which would allow them to acquire more cultural capital pertaining to their degree of expertise on the MKX application.

5.5. The Sao Paulo Team Field

The team members at C1 from Sao Paulo belonged to a nested field labelled as *Sao Paulo Team,* with its own set of relevant symbolic and cultural capital. No form on economic capital was identified for this sub-field. This sub-field was characterised by one position namely: *Brazilian Software Developers.* The position, habitus and work practices the *Brazilian Software Developers* are described below.

5.5.1. Brazilian Software Developers' Work Practices

In order to describe the circuit of reproduction of the *Brazilian Software Developers'* work practices, each element in the Figure 5.6 is described.

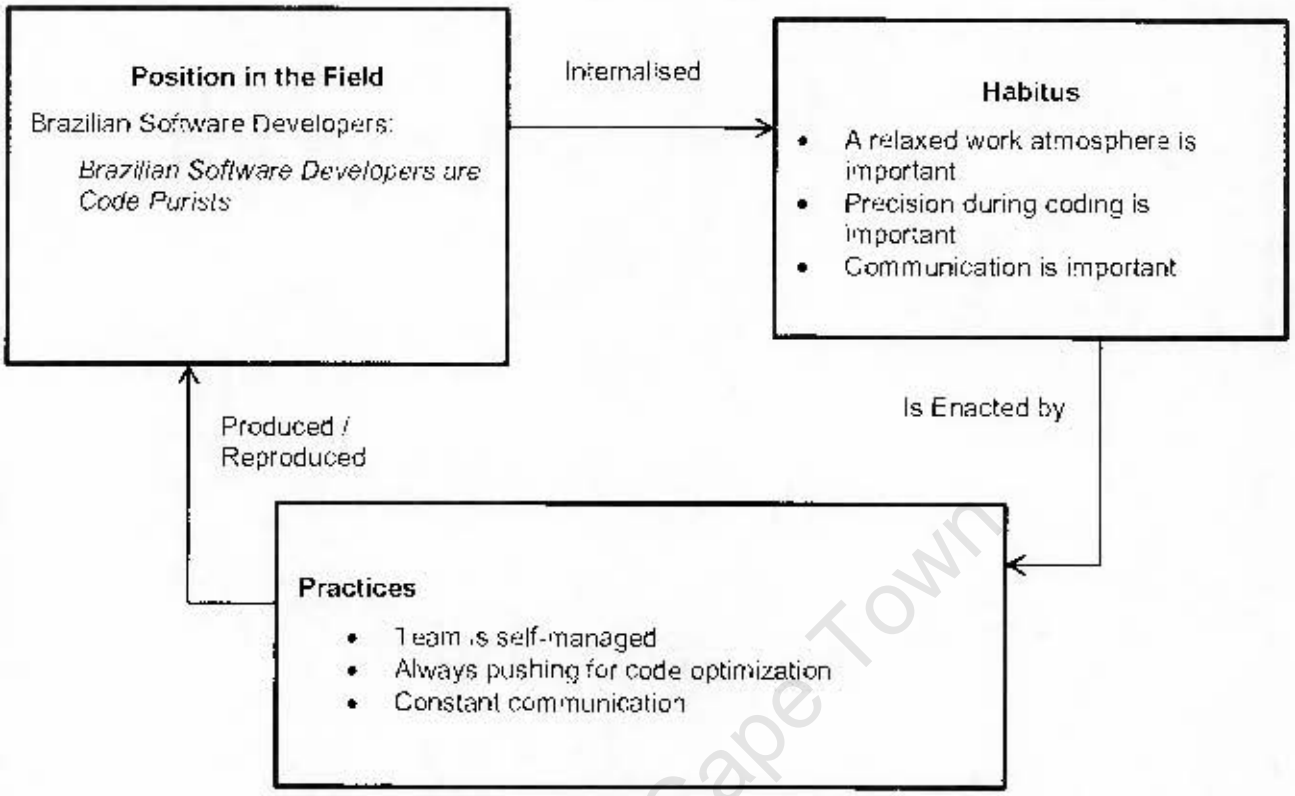


Figure 5-6 – Brazilian Software Developers' Work Practices

5.5.1.1. Brazilian Software Developers Position in the Field

The *Sao Paulo Team* field was composed of three agents, two of whom were interviewed, namely C1.R6 and C1.R7. They were all expert in one specific programming language. C1.R6 was in a dominant position while C1.R7 was in an equivalent position. The difference in their respective positions of power can be attributed to the difference in cultural capital which they possessed.

Agent	Knowledge of Programming Language	Expertise in Software Development Tasks
C1.R6	Expert in PHP	Experienced in distributed Scrum prior to joining the C1 GASD team. Expert in back-end software development using PHP Able to perform any programming tasks required of him in addition to PHP
C1.R7	Highly experienced in Javascript and CSS, but was only recently introduced to PHP at the time of the study	Highly experienced in front-end software development. Newly introduced to PHP back-end software development tasks

Table 5-6 – Sao Paulo Team Field Cultural Capital

As can be seen in Table 5.6, C1.R6 was an expert in PHP and held high degree of expertise in completion of general software development tasks. C1.R7 was a novice in the PHP language and her expertise was only relevant to front-end software development. The expertise in the PHP language was highly valued in that field.

Also, upon examining the relevant forms of symbolic capital for the *Sao Paulo Team* field, it could be seen that the seniority levels of the agents (symbolic capital) was important; while C1.R6 was a *senior* software developer, C1.R7 was not (see Table 5.7). The agents belonging to the *Brazilian Software Developers* position were perceived as being code purists.

Agent	Seniority Level
C1.R6	Senior Software Developer
C1.R7	Software Developer (not senior)

Table 5-7 – Sao Paulo Team Field Symbolic Capital

5.5.1.2. Habitus and Practices of the Brazilian Software Developers

The habitus internalised by the *Brazilian Software Developers* related to the fact that they valued a relaxed work atmosphere, planning, precision during coding, and communication. These are illustrated in Figure 5.6 and are further described below.

- *A relaxed work atmosphere is important*

Brazilian Software Developers valued the feeling of freedom and being relaxed in their workspace. They explained that they valued the fact that *they were not in the same space as the bosses* (C1.R6) and because of that, felt that their productiveness was leveraged. It was important for them not to be dictated on how to work and they valued the feeling of not having any rules in place within their workspace:

> *It's not like: I have to work in this way or that way. It felt better because we didn't have rules set in place. It felt like how you would work if you were not working in a team* (C1.R6)
> *Personally for me I work in a much more relaxed way, much more productive* (C1.R7)

The practices which they were predisposed to engage in given this form of habitus are further described below.

- ***Team is self-managed***

From the interviews, it came across that it was important for the Brazilian team members to be self-managed. The *Brazilian Software Developers* valued this practice as it made them feel free. The work hours for the Brazilian team members also appeared to be quite flexible and they often started work later during the day than the normal starting time and left later than the normal ending time at night: "*we will typically come to work at 9am and work till 6pm, over there they don't do that. They'll come at 10am or 11am, and they work till very late, maybe till 8pm, 9pm at night*" (C1.R8). They also worked extra hours willingly and spontaneously whenever required:

> "*it feels more natural for the team to work extra hours without being asked to. It's either I'm going to be sitting in the car, or I'm going to be sitting here doing something. So we tend to sit in the office. Not all of us, for example me, I don't use a car so I don't have that kind of limitations. But usually we stay in the office after 6pm*" (C1.R6).

These work practices served to reinforce the structure of the *Sao Paulo Team* field.

- *Precision during coding is important*

The *Brazilian Software Developers* felt that it was important to be as precise as possible while coding. They strived to abide by best coding practices (particularly relating to the PHP programming language), which they identified by participating in coding conventions, forums and conferences:

> "*Historically, the Brazilian team members have worked more on the back-end component. And in the backend component, it's very much common to work in a precise fashion. They focus on coding procedures and best practices and so forth*" (C1.R3).

The practices which they were predisposed to engage in, given this habitus, are further described below.

- ***Always pushing for code optimization***

The work practice that the *Brazilian Software Developers* were predisposed to engage in, given their habitus valuing a relaxed work atmosphere, was the constant push for opportunities to work on code optimization. Team members from South Africa described the *Brazilian Software Developers* as "*code purists*" because of this. This practice was

relevant to the *Sao Paulo Team* field but also impacted on the overall interaction between the team members of the C1 GASD team during the sprint planning meetings where tasks' duration had to be established. In particular, for code optimization to be performed tasks' duration was expected to be longer, which was not always approved by all team members. This will be further elaborated in Chapter Seven.

- *Communication is important*

The need to maintain constant and open communication amongst all the team members was highly valued by the *Brazilian Software Developers*. Given the fact that English was not their first language, they felt that communicating as often as possible and through various communication channels would maximise their chances of putting their point across to the rest of the team. *Brazilian software developers* described themselves as being *much more defensive about [their] point of view* and explained that they *put a lot personal feelings into their work* (C1.R6). Consequently, the need to communicate was highly valued, to allow them to express their opinion and to be heard. The practices which they were predisposed to engage in given this habitus are further described below.

- ***Constant communication***

The *Brazilian Software Developers* reported that because of geographical distance and language differences, they sometimes had difficulties in fully expressing themselves and were not always understood by the South Africans: *"English is not my first language and I am not really fluent in it. So I'm not always saying all the things I want to say."* (C1.R7). This situation could compromise their position within the larger C1 GASD field, and they therefore strategized (without any genuine strategic intent) to ensure that they were equally able to voice their opinion within the team. The strategy involved communicating as often as possible through as many communication channels as possible:

> *"English is not the mother tongue language for any of us. And at times we don't know how to express ourselves in English. We might miss a word here and there. It's not really major but every now and then we have to stop and think about the word that we are looking for. We are overcoming that and we strive to communicate as often as possible. Register everything, send everything by email, when someone does something we let the whole team know. Just keep all the information on the table as much as possible"* (C1.R6).

However, the South African team members from the C1 GASD team felt that the Brazilians were at times abrupt in their communication: *"the language difference is also a problem. For example, sometimes they'll type us an email. It will sound very funny to us, sometimes I've taken it to be very arrogant, and when you speak to them in person, it's not like that at all. And this is because their English is not as good as can be"* (C1.R8).

All three forms of practices that the *Brazilian Software Developers* were predisposed to engage in served to reproduce the structure. In particular, while strategizing to be self-managed, they had enough space and time to decide to implement particular coding practices (provided that they did not go over the time available to complete a user-story). The practice of pushing for code optimization, allowed them to defend their position as code purists, and while constantly communicating, they strived to entice the rest of the team to embrace their views about best coding practices.

5.6. The Brazilian PHP Software Developers Field

Some of the team members, based in Sao Paulo, also belonged to a separate field labelled as the *Brazilian PHP Software Developers* field, with its own set of relevant symbolic and cultural capital. Similarly to the *Sao Paulo Team* sub-field, no form of economic capital was identified for this sub-field. The *Brazilian PHP Software Developers* sub-field was characterised by one position, namely: *PHP Experts*.

5.6.1. PHP Experts' Work Practices

To describe the circuit of reproduction of the *PHP Experts'* work practices, each element in Figure 5.7 is described.

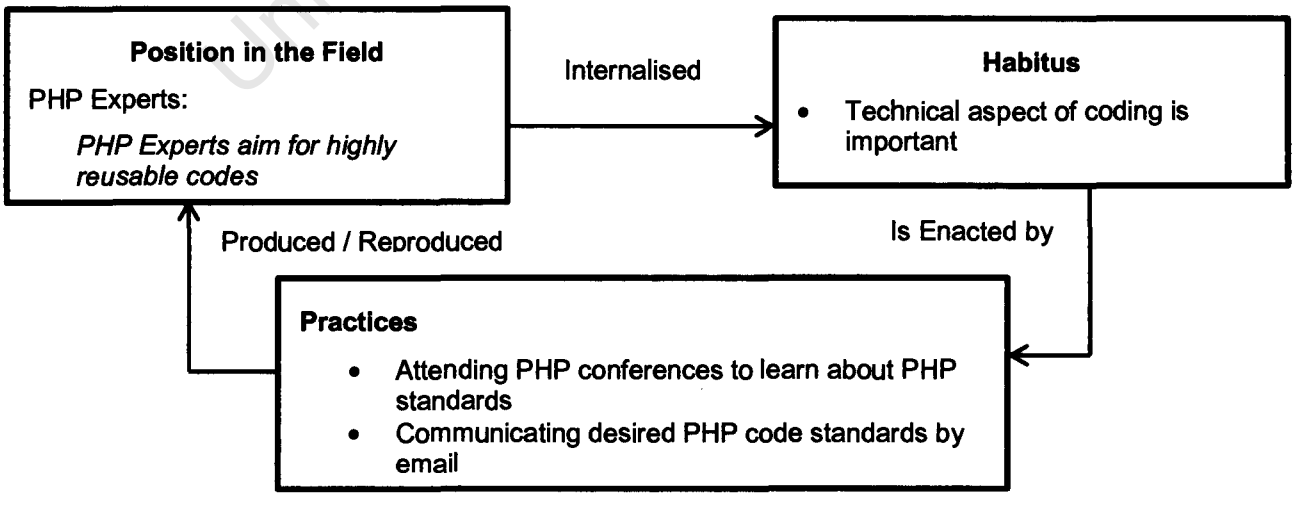


Figure 5-7 – PHP Experts' Work Practices

5.6.1.1. PHP Experts Position in the Field

The *Brazilian PHP Software Developers* field was composed of two team members, C1.R6 and C1.R7, who both shared the position of *PHP Experts* in that field. C1.R6 and C1.R7 held a position of dominance and subordination respectively in the *Brazilian PHP Software Developers* field. C1.R6 was in a dominant position because of his high amount of cultural capital pertaining to his skills in the PHP language (see Table 5.8) and his symbolic capital pertaining to a high degree of influence within the overall PHP community (see Table 5.9). C1.R7 was a novice and was newly acquainted with the PHP language (see Table 5.8). She was therefore newly involved in the wider PHP community (see Table 5.9).

Agent	PHP software development skills
C1.R6	Highly skilled and experienced in PHP.
C1.R7	Low level of skills in PHP software development

Table 5-8- Cultural Capital for the Brazilian PHP Software Developers Field

Agent	Degree of Influence within the overall PHP community
C1.R6	Highly influential and often gives lectures at PHP conferences
C1.R7	Newly involved within the PHP community

Table 5-9 - Symbolic Capital for the Brazilian PHP Software Developers Field

5.6.1.2. Habitus and Practices of the PHP Experts

- *Being output oriented is important*

As part of their habitus, the *PHP Experts* were predisposed to value the technical aspect of coding. In particular, they believed that any algorithm being written should be optimised in order to achieve reusability and low execution time: *"basically I would like to see more focus on the technical side. Some issues on the code, stuff that needs to be reworked [...]. We do have stories where we need to refactor the code of features that are already working, but make it better"* (C1.R6). They thus perceived themselves as the only ones focused on code optimization and reusability in the team and were not always willing to compromise during sprint planning. This form of habitus was formed out of socialization processes with members of the wider PHP community and past experiences in other PHP related projects. The practices which they were predisposed to engage in, given this form of habitus, are further described below.

- ***Attending PHP conferences***

The *PHP Experts* within the *Brazilian PHP Software Developers* field often participated in international PHP conferences, where they acquired knowledge about the latest PHP standards and best practices: *"the Brazilian PHP developers attend conferences on PHP and are very much on "avant-garde" concerning how to code in PHP"* (C1.R2). While this work practice was in line with the habitus of focusing on the technical aspect of coding and served to reproduce the structure, it was also a strategy employed by the experts to defend their position in the *Brazilian PHP Software Developers field* as well as the *C1 GASD* main field. In particular, by attending these conferences, they acquired more cultural capital, allowing them to defend their positions as experts within both fields.

- ***Communicating the required code standard by email***

Given their habitus of favoring the technical aspect of coding to achieve reusability and fast execution time, the *PHP Experts* were predisposed to communicate their expectations of the code standards for the MKX application to the rest of the C1 GASD team: *"I usually send email detailing the standards that I'm looking at, this is what we should be doing, this is how it affects our work" (C1.R6).*

Similar to the practice of attending PHP conferences, the act of communicating the required code standard by email can also be perceived as a double-meaning strategy. At first sight, it might appear as if this practice was meant to enhance the code standard of the MKX application and served to reproduce the structure of the field. However, it also served to enhance the Brazilians PHP software developers' position within the *Brazilian PHP Software Developers* field and the *C1 GASD* field.

5.7. Chapter Summary

In this chapter, the various overlapping fields and sub-fields identified for the C1 case study have been described as well as their relationships with each other. In particular, five key fields were found relevant to C1 namely the: *C1 GASD Team* Field, *Software Development* field, *Cape Town Team* field, *Sao Paulo Team* field, and *Brazilian PHP Software Developers* field. The circuits of reproduction of the work practices prevailing in these fields and sub-fields have been described, paying particular attention to the positions, habitus, practices and forms of cultural, symbolic and economic capital. Information provided from this chapter will be used as a basis to describe and explain the Scrum process breakdowns identified in C1 in Chapter seven.

6. EMPIRICAL OBSERVATIONS FOR DESIGN CASE (CASE 2)

The second case study (C2) undertaken for this research project is in line with Bonoma (1985)'s "design" stage of case study design. This chapter details the findings uncovered from the analysis of the data, and particularly focuses on the circuit of reproduction of the work practices at play within C2. The chapter is structured in six sections. In Section 6.1, an overview of the various overlapping, joint and nested fields identified for C2 and the relationships between them is presented. In Sections 6.2 to 6.5, the fields are described, with particular attention paid to the positions in the field, habitus, the work practices and the relevant forms of cultural, symbolic and economic capital. The chapter is concluded in Section 6.6

6.1. Overview of Fields Identified in C2

Four key fields were identified for C2: *C2 GASD Team, Pune Team, Durban Team,* and *Sanbi (C2 Organisation)*. Similarly to C1, these fields in C2 were also influenced by larger fields relevant to the wider software development community namely: *GSD Community,* the *Agile Community* and the *Scrum Community*. This is illustrated in Figure 6.1.

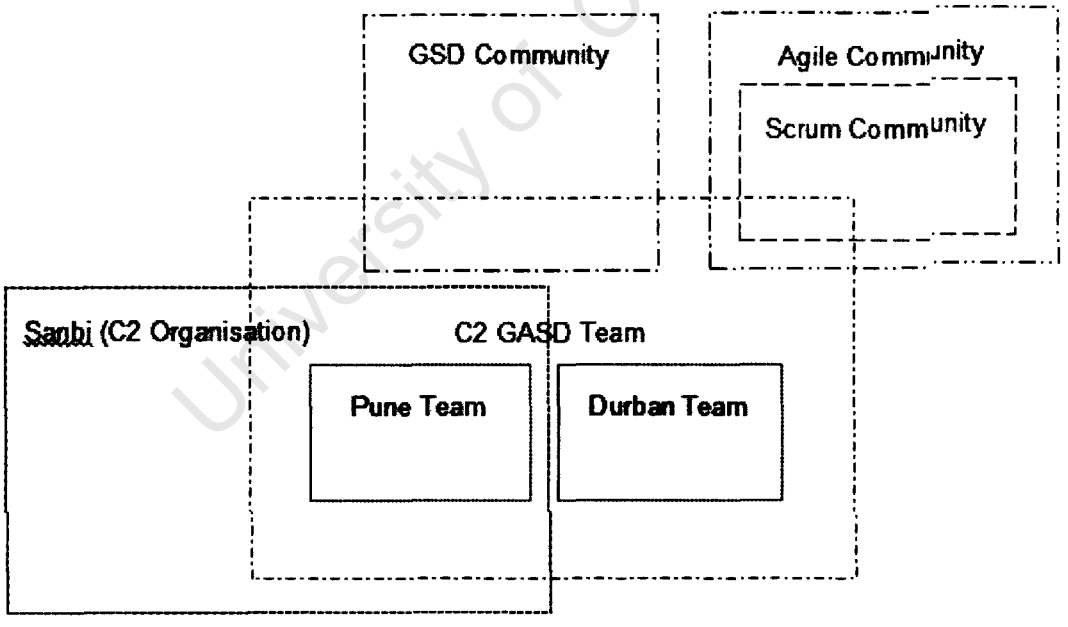


Figure 6-1 – Overview of C2 Fields: Overlapping, Joint, and Nested Fields

It can be seen in Figure 6.1 that, given the dispersed geographic locations characterising the software development context at C2, the *C2 GASD Team* field is composed of two sub-fields namely *Pune Team, Durban Team.* The *C2 GASD Team* field was also influenced by the *GSD Community,* the *Scrum Community* and the *Agile Community* fields, given the nature of their work at C2. Similarly to C1, the *Agile Community* and the *GSD Community* fields further formed part of the *Software Development Community* field, which has not been illustrated in Figure 6.1, in order to retain the focus on the case and agile software development.

The circuit of reproduction of the work practices at play in the four main fields, *C2 GASD Team, Pune Team, Durban Team, Testers* and *Sanbi (C2 Organisation),* will be described in the following sections. The different forms of relevant capital (cultural, symbolic and economic) operating in them will also be detailed.

6.2. The C2 GASD Team Field

The *C2 GASD Team* field is characterised by four key positions which form the structure of the field: *Project Managers, Testers, Customers and Software Developers.* The circuit of reproduction for the Project Managers and Testers' work practices will be described in Sections 6.2.1 and 6.2.2. *Customers* and *Software Developers* work practices circuit of reproduction will be described in sections 6.3 and 6.4 as they also form part of nested fields with distinctive forms of capital which will be discussed separately.

6.2.1. Project Managers' Work Practices

To shed light on the work practices of *Project Managers* in the *C2 GASD Team* field, each element in Figure 6.2 is described.

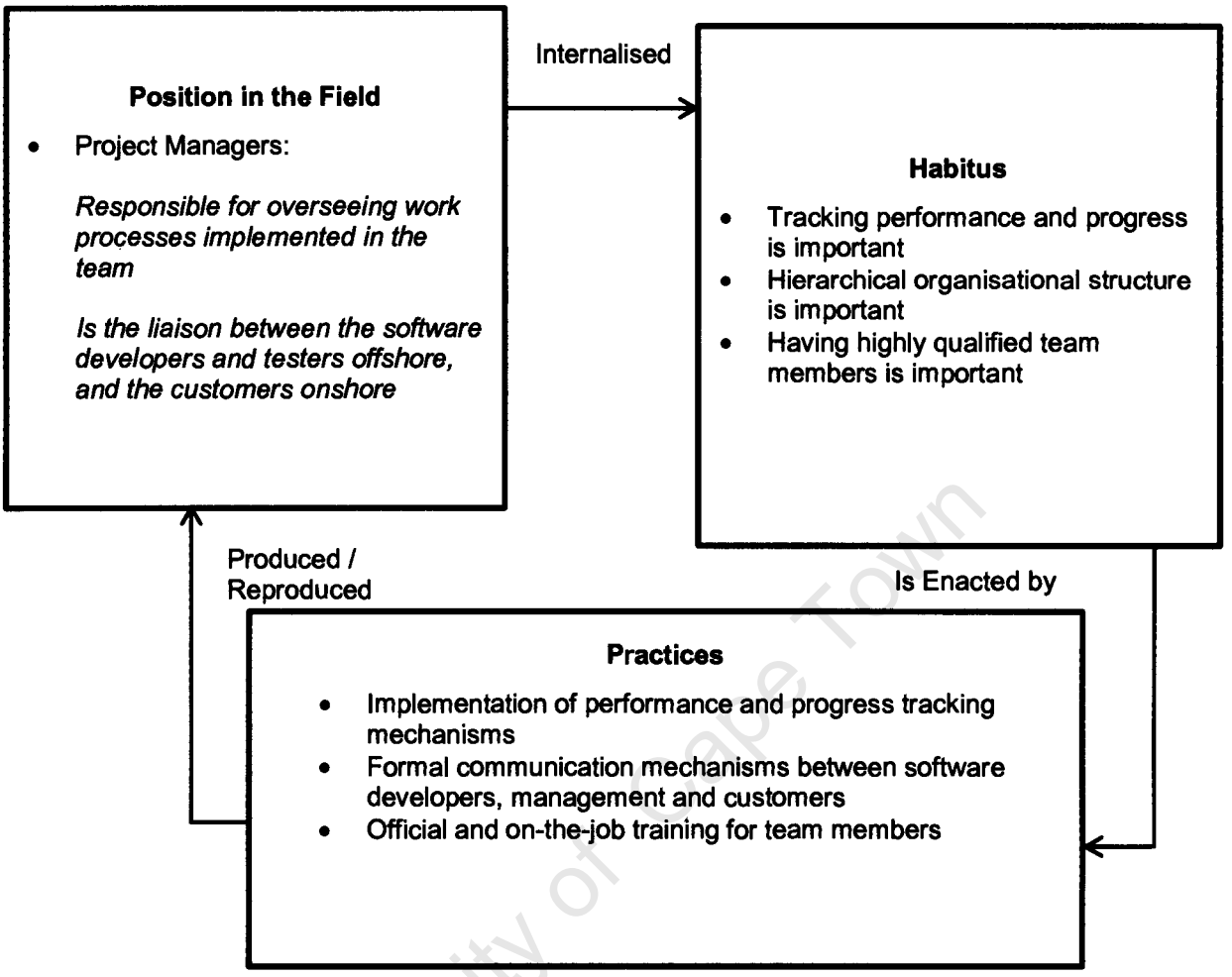


Figure 6-2 – Management's Work Practices

6.2.1.1. Project Managers' Positions in the Field

An agent in the *Project Manager* position in the *C2 GASD Team* field was responsible for overseeing the work processes implemented and followed in the team:

> *If there are certain processes outside the project which need approval, [management] is here. Things like getting software, getting hardware. In such cases you do need your manager* (C2.R9).

In addition, this agent would act as liaison between the software developers and testers offshore and the customers onshore, on project management and offshore team

members' performance matters: *It is my job to give suggestions. During discussion, both parties give their own suggestions. And what we mutually agree is implemented* (C2.R8).

C2.R8 was the only agent in that particular position in the *C2 GASD Team* field. He also was in a dominant position in the field. Three forms of cultural capital, of which he possessed high amount, allowed him to be in this dominant position (see Table 6.1). He had two years of experience in the project team as he had been working in the team since its inception. He also had a high degree of experience in the Scrum methodology and project management in general. In addition, even though his technical expertise in the Colin application was low, he had a good understanding of the business logic of that application, which allowed him to monitor the productivity of the offshore software developers. He also held a high amount of symbolic capital. In particular, given his previous experience in other projects at Sanbi, he was promoted to a senior position and was given the job title of Project Manager, which contributed further towards giving him a dominance position in the field.

C2.R8 defended his dominant position in the field by ensuring that all requests from the software developers pertaining to the Scrum process were always passed on to him, prior to being forwarded to the customers onshore. He justified his actions by claiming that the software developers were too busy with development work to worry about meeting with customers to discuss administrative matters: *Actually I take input from them because everybody [software developers] here is so busy in development and they don't have time to go to such meetings. That's why they give their inputs to me and I share the minutes of meetings with them* (C2.R8).

Agent	Number of Years of Experience in Project Team	Experience in Agile Software Development	Current Project Experience
C2.R1	1.5 months	Experienced in agile software development as she has been working on Scrum projects in C2 for the past 3 years and 5 months	Limited experience as a tester for that given project, as she only recently joined the team
C2.R2	1.5 months	Experienced in agile software development and executed a case study for C2 on various possible agile methodologies which could be used prior to Scrum implementation in the team	Limited experience as a tester for that given project, as he only recently joined the team
C2.R3	2 years	Attended Agile training and has been involved with Scrum since the beginning of the project	Highly experienced in the business logic of the Colin application, as he has been working on the project since the beginning
C2.R4	2 years	Never attended any agile training and has 2 years of experience with the use of Scrum (only with the current project)	Highly experienced because of good knowledge of the project and had also worked as onsite coordinator for 1 year
C2.R5	1 year	First project experience in agile software development. Graduated from University 1 year ago.	Limited experience in the Colin application project
C2.R6	2 years	Experienced in agile software development prior to joining C2.	Highly experienced in business logic and architecture of Colin application, as he has been working on the project since its beginning
C2.R7	3 weeks	Attended agile training in South Africa and has been involved in other Scrum projects in other teams at C2	Limited experience in the Colin application, as she only recently joined the team.
C2.R8	2 years	High degree of experience in Scrum and project management in general	Limited technical experience in the Colin application but

			understanding of the business logic
Casper	2 years	Good Experience in Scrum and has been involved in Agile in previous projects at the customer site	High degree of experience because of excellent technical knowledge of the Colin application. Acting as Scrum Master and Product Owner. He was also the project initiator for the Colin project
Yogi	2 years	Good Experience in Scrum and has been involved in Agile in previous projects at the customer site	High experience in the Colin application given his role as technical leader and onsite coordinator
Phillip	2 years	Good Experience in Scrum and has been involved in Agile in previous projects at the customer site	High experience in the business logic of the application. Acting as Product Owner for the Colin application.
Jarred	1.5 months	Only joined the project when the Quality Assurance (QA) team was introduced to the Colin project team. Previous experience in Scrum in other projects at the customer site	Limited experience as a tester for that given project, as he only recently joined the team

Table 6-1 - Cultural Capital for C2 GASD Team Field

Forms of cultural capital relevant to the *C2 GASD Team* field are described in Table 6.1. As previously mentioned, these relate to the agents' number of years of experience in the project team, their experience in agile software development, and their current project experience. The table identifies the forms of capital owned by each member of that field. The agents labeled as *Casper, Yogi, Philip,* and *Jarred* were named differently to the others (e.g. C2.R1) because they were not respondents from the study, even though they still belonged to the *C2 GASD Team* field.

Agent	Job Title
C2.R1	Tester
C2.R2	Senior Tester
C2.R3	Team Leader
C2.R4	Software Developer
C2.R5	Software Developer
C2.R6	Senior Technical Specialist
C2.R7	Senior Software Developer
C2.R8	Project Manager
Casper	Scrum Master / Project Manager
Philip	Product Owner
Yogi	Onsite Coordinator
Jarred	Tester

Table 6-2 - Symbolic Capital for C2 GASD Team field

The form of symbolic capital relevant in the *C2 GASD Team* field is presented in Table 6.2 and related to the agents' job title. The economic capital relevant to that field was the degree of control in the Colin application project and has been described in Table 6.3. It can be seen in that table that the only agents possessing some form of economic capital were Casper and Philip.

Agent	Ownership of project
C2.R1	Low
C2.R2	Low
C2.R3	Low
C2.R4	Low
C2.R5	Low
C2.R6	Low
C2.R7	Low
C2.R8	Low
Casper	High
Phillip	Medium
Yogi	Low

Table 6-3 - Economic Capital for C2 GASD Team field

6.2.1.2. Habitus and Practices of Project Managers

The various habitus internalised by *Project Mangers* related to how the team valued: the tracking of performance and progress, a tall organisational structure, meeting the demands of the customers and producing highly qualified team members.

- *Tracking performance and progress is important*

Project Managers valued tracking of the progress and performance of team members, particularly of the software developers at the offshore site. Furthermore, *Project Managers* expressed a desire to have a detailed view of the time spent in completing user stories and the productivity level of each team member during a sprint. The importance of tracking performance and progress was visible through the various mechanisms put in place. For instance:

> *Each task is stereotyped, like development, unit testing, code review, test script writing and things like that. These are included as tasks. We include some time for design and some time for design review as well* (C2.R8).

By breaking down a user story into specific *stereotyped tasks*, the time spent by software developers or testers in completing their tasks could be monitored and their progress and performance recorded. *Project Managers* implemented various mechanisms within the team in response to the predispositions derived from this habitus.

- ***Implementation of performance and progress tracking mechanisms***

One of these mechanisms was the review meetings which *Project Managers* had with the customers onshore, to discuss the performance of the software developers and testers, as well as code quality:

> *In that review meeting, we talk about code quality, i.e. if there are any issues pertaining to code quality. Also, if the productivity of one particular team member is low, we talk about it. So basically we discuss around these two points. The productivity determines the speed or velocity. Code quality is basically the amount of rework. The number of defects will be a measure of code quality according to the client. And that will impact on rework* (C2.R8).

For progress tracking purposes, *Project Managers* recorded information about the progress of the software development and testing tasks on a daily basis. The information

was later used to track the velocity of the team on a burn-down chart. In the team, velocity was described as: *whatever number of stories we have initially planned to complete, and against that, how many we have completed* (C2.R8). Velocity was important not only to *Project Managers* but also to the customer, as a reassurance that the work was being done and that progress was being made by the team. Consequently, in providing the customers with a constant visibility of the team's velocity, *Project Managers* reinforced their position as liaison between the offshore team and customers.

The tracking of progress and performance was also managed through the use of a wiki. In particular each user story scheduled for completion during a sprint was recorded in a wiki as well as the stereotyped tasks associated to it. As noted in the following extract, the name of the software developer or tester to which the task was allocated was also recorded:

> *In the wiki we also have a break down. The analysis part is there, the design part is there, the UI part is there, coding part is there and testing part is there for a feature. The basic feature is given e.g. login implementation and then the sub task. Then we allocate the hours and we track who is doing what* (C2.R4).

Project Managers also carefully monitored the extent to which the requests put forward by the customers were being executed by the software developers and the testers. On a daily basis, the manager would walk around the office space and personally chatted to each software developer and tester, gathering feedback on how their work was progressing since the previous day. He asked specific questions on the issues which were raised by the customers onshore during the previous daily stand-up meeting and recorded their responses. These were then compiled in a report which was passed on to the customers during his private communications with them. He explained that in doing so, he ensured that the team members focused their attention on the right tasks which he felt would meet the customer's demands.

The final mechanism introduced by *Project Managers* to track the performance level of the C2 GASD Team was the use of an onsite coordinator. *Project Managers* agreed to the introduction of an onsite coordinator for a different reason to that of the customer (to be discussed later). For *Project Managers*, the onsite coordinator would ensure that the offshore team obtained the right requirements from the customer. It would allow them to better understand the requirements and have someone onshore to discuss the feasibility

of some requirements from a technical perspective. This would ensure that the offshore software developers would obtain precise and feasible requirements, without which their productivity would be hindered:

> *We should know what exactly the problems are, what exactly the client means and whether that feature is technically feasible or not, and the onsite coordinator helps us to do that* (C2.R8).

While the strategic intent for implementing the onsite coordinator role appeared to address the need to boost the chances of the offshore team to deliver functionalities, meeting the customer requirements and thus maintain a high performance level, this work practice also served to reinforce the position of Project Managers in the structure. In particular, by introducing the onsite coordinator as part of the work process, *Project Managers* reinforced its role as process coordinator, and reasserted its dominant position pertaining to being in charge of the work process update in the structure.

Overall, *Project Managers* justified tracking the team's progress and performance level by the need to provide a good service to the customers onshore. However, this strategy served to reinforce their position in the *C2 GASD Team* field. Through these practices, *Project Managers* reasserted the importance of its role in the team by positioning itself as the coordinator of all practices geared towards tracking the team's progress and performance. Consequently, their dominant position in the team was maintained.

- *Hierarchical organizational structure is maintained*

Project Managers orchestrated a habitus whereby a hierarchical organizational structure, as opposed to a flat structure, was valued. This habitus could be identified from the formal manner in which software developers and the testers interacted with *Project Managers*. The software developers were restricted in the way they communicated with top Project Managers, as shown from this extract:

> *Here we are very friendly, but some kinds of restrictions are there according to the management different levels. If [the manager] is there, I cannot just say anything in front of him. I have freedom, but before saying anything, I should think first. We are friendly, but we have some kind of hierarchy and we maintain the hierarchy levels* (C2.R1).

Given habitus, the practices which Project Managers was predisposed to engage in are described below.

- ***Formal communication mechanisms between software developers and testers, Project Managers and customers***

Given the habitus valuing a hierarchical organisational structure, *Project Managers* was predisposed to follow certain work practices. Firstly, formal communication mechanisms were put in place between the software developers and testers, Project Managers and customers. For example, when software developers had to express grievances or issues which they were experiencing, they had to send official emails to *Project Managers* to request a meeting for further discussion on the matter, as can be seen in that extract: *I send an official email saying that I want to talk to him. And then I meet him personally in the meeting room and then I can explain those things* (C2.R4). Any managerial issues concerning the customer had to be passed onto *Project Managers* who then decided whether they were valid enough to be presented to the customer.

Some team members were excluded from retrospective meetings, which then involved only the customer onshore and Project Managers offshore:

> *Our manager C2.R8 is having weekly meetings with the client. It's kind of a retrospective meeting. It's not sprint retrospective but it has the same purpose as a retrospective meeting. The kinds of things discussed in the meeting are: what are the areas for improvement, what are the lessons learnt* (C2.R3).

Project Managers justified this practice by stating that the exclusion of software developers and testers from the meetings was specifically requested by the customers. However, *Project Managers* also adhered to this practice as it served as a double-meaning strategy, allowing them to maintain their dominant position in the field as liaisons between the offshore software developers and testers and the onshore customers.

In spite of these circumstances, the software developers and testers still believed they could express their views to the customers on how to improve the Scrum process by passing them onto *Project Managers*, as can be seen through this extract:

> *One day before the meeting, (C2.R8) comes to us and asks us if there are any improvements from the team side, or any problems or bottlenecks. For example in the last two months we needed a staging DB. So we put that forward to C2.R8 and in that meeting* [C2.R3].

However, the suggestions provided by the team, were not always passed on to the customer. These suggestions were instead filtered by C2R8 who considered it his role to determine which suggestions were worthy. At times, the team only reported the challenges which they faced, and left it to C2.R8 and the customer to come up with a solution: *Yes, if we have a problem we can tell [C2.R8] and his job is to resolve it* (C2.R5).

- *Having highly qualified team members is important*

The importance of having highly qualified team members was valued by *Project Managers* in the *C2 GASD Team* field. By leveraging the team members' level of expertise and qualification, *Project Managers* believed that it would provide incentives for members to remain loyal to the team and to Sanbi. For example, when agile was first introduced, an external organization, expert in the use of agile methodologies, was hired to evaluate the suitability and effectiveness of the Scrum work practices in place in the C2 GASD team and in Sanbi as a whole. In addition, this expert was also tasked to assess the expertise level of the team members on Agile and Scrum. It was found that the software developers did not have a good knowledge of Scrum and what it entailed to be agile:

> *They basically tracked four parameters: processes, people, communication and environment. For the people aspect, what was found is that [the team] did not have theoretical knowledge of Scrum. The training was done on that so that they know what it means* (C2.R8).

Measures were then taken to leverage the team members' skills accordingly, and are further described below.

- ***Official and on-the-job training for team members***

In light of their habitus valuing the need to have highly qualified team members, *Project Managers* were predisposed to organize training sessions on agile and Scrum to the team

members, whenever they lacked theoretical knowledge on that topic. In addition to that, *Project Managers* were also predisposed to organize both official and on-the-job training, on the Colin application, to the software developers and testers. The software developers reported that whenever they wished to acquire knowledge on a particular type of technology, they could always inform *Project Managers* about the matter and they would be sent on training provided that it did not impact on their workload and ability to deliver their tasks on time:

> *If we are going to work on new technologies, we can ask the managers for extra training* (C2.R4). When new members joined the team, they would receive a *combination of official corporate training, as well as on-the-job training* (C2.R5).

6.2.2. Testers' Work Practices

To shed light on the work practices of *Testers* in the *C2 GASD Team* field, each element in Figure 6.3 is described.

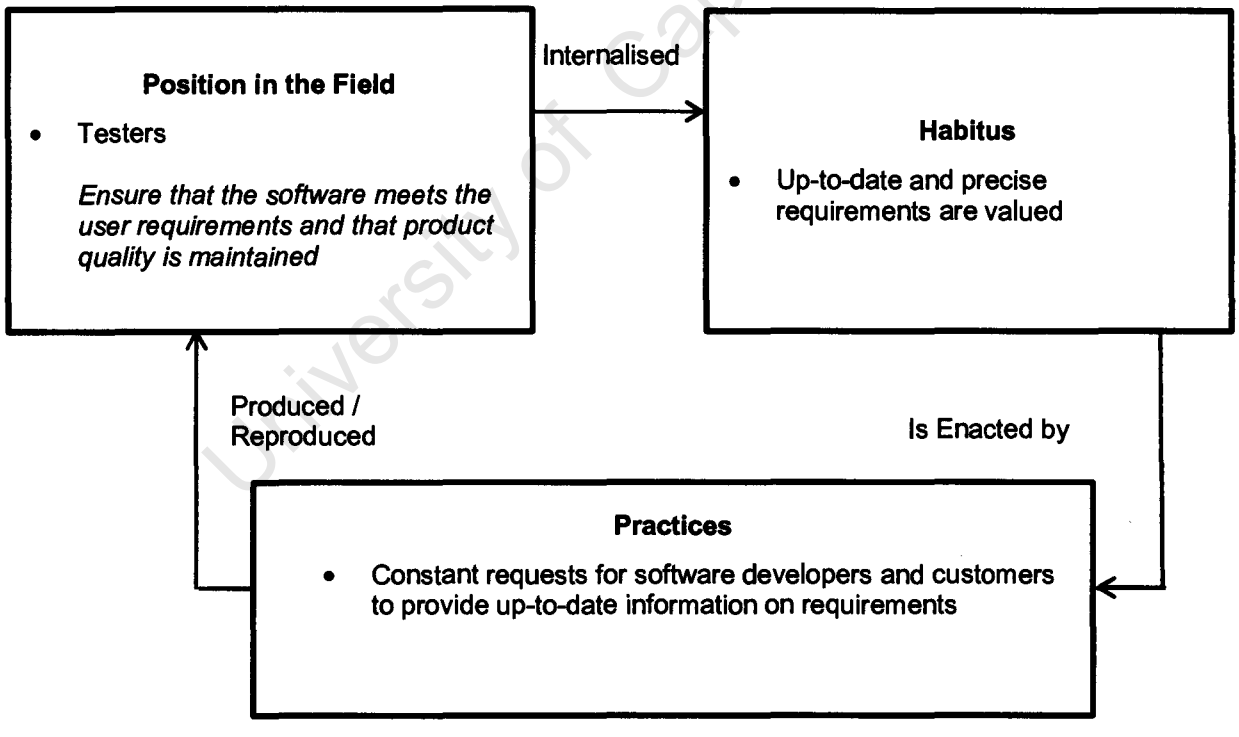


Figure 6-3 – Testers' Work Practices Circuit of Reproduction

6.2.2.1. Testers Positions in the Field

Agents sharing the *Testers* position in the *C2 GASD Team* field were the ones responsible for ensuring the quality of the Colin application, in terms of the extent to which the customers' requirements were being met: *The testing team is responsible for clarifying requirements, asking and understanding all testing feasibilities, scenarios (TDD) and test execution of previous sprints* (C2.R2).

From the inception of the project to the end of sprint five, there were no quality assurance (QA) tasks included in the task breakdown of each user story and hence, no testers were present in the team. The *Testers* were only introduced in the *C2 GASD Team* field from sprint six. Three agents from both India (Pune) and South Africa (Durban) occupied the *Testers* position in the field namely: C2.R1, C2.R2, and Jarred.

All three agents held equivalence positions in the *C2 GASD Field* despite their limited amount of cultural capital relevant to that field. In particular all three agents had only been part of the team for the last 1.5 months (since the beginning of sprint 6), and thus had limited experience in and knowledge of the business logic of the Colin project. However, they were all experienced in the application of Quality Assurance in Scrum, given their participation in other Scrum projects in C2 (see Table 6.1). Pertaining to their symbolic capital, C2.R1 and Jarred were "testers" and C2.R2 was a "senior tester" (see Table 6.2).

6.2.2.2. Habitus and Practices of Testers

The analysis identified one form of habitus which served as the generative basis for the work practices of the *Testers*. The form of habitus is:

- *Up-to-date and precise requirements are valued*

The *Testers* demonstrated a strong belief in the need to obtain clear, precise, and up-to-date user requirements from the specification documents. The quality of information contained in these documents was crucial for them to compile accurate test cases and successfully complete their testing tasks. Testers often made statements in the line of:

> *There should be proper documentation for everything, all the requirements etc.* (C2.R1) or *I think that here there is not enough documentation. They have proper requirements, but it's not detailed enough* (C2.R2).

Some even justified their beliefs by stating that: *According to agile, there should be documentation* (C2.R2). By aligning their values to what they thought was specified by the Scrum methodology, they sought to formalize the process of clearly eliciting requirements in the user requirements specification documents. The various practices to which the *Testers* were predisposed to engage in are described below.

- ***Constant requests for software developers and customers to provide up-to-date information on requirements***

The *Testers* constantly strived to encourage all the other team members to provide them with up-to-date information on the user requirements, which they felt were never clear and precise enough. They tried to overcome these challenges by *cross-checking the user stories* (C2.R2) and by getting as involved as possible in the requirement analysis stage, as shown in this extract:

> *During the requirement analysis phase, the testing team gets involved and we have questions and answers sessions* (C2.R2).

They also constantly asked for clarification from the business analyst or the person who wrote the user stories:

> *If I have some doubts about requirements, I can always ask (C2.R4). If the requirements are not clear, I can send the queries to BA. So it's all written communication, or calls. So when we have a clear understanding of the requirement, then we implement the coding and the test cases* (C2.R1).

The *Testers* engaged in these practices to successfully complete the tasks required of them (i.e. maintaining the quality of the Colin application project) and, in doing so, reinforced the structure. But at the same time, they also sought to acquire more cultural capital in terms of the current project experience, which would serve to improve their position in the *C2 GASD Team* field.

6.3. The Pune Team Field

The team members at C2, based in Pune, were seen to belong to a separate nested field labelled as *Pune Team.* The *Pune Team* field had one position namely: *Software Developers.* The position, habitus, and work practices the *Software Developers* are described in section 6.3.1.

6.3.1. Software Developers' Work Practices

To shed light on the work practices of *Software Developers* in the *Pune Team* field, each element in Figure 6.4 is described.

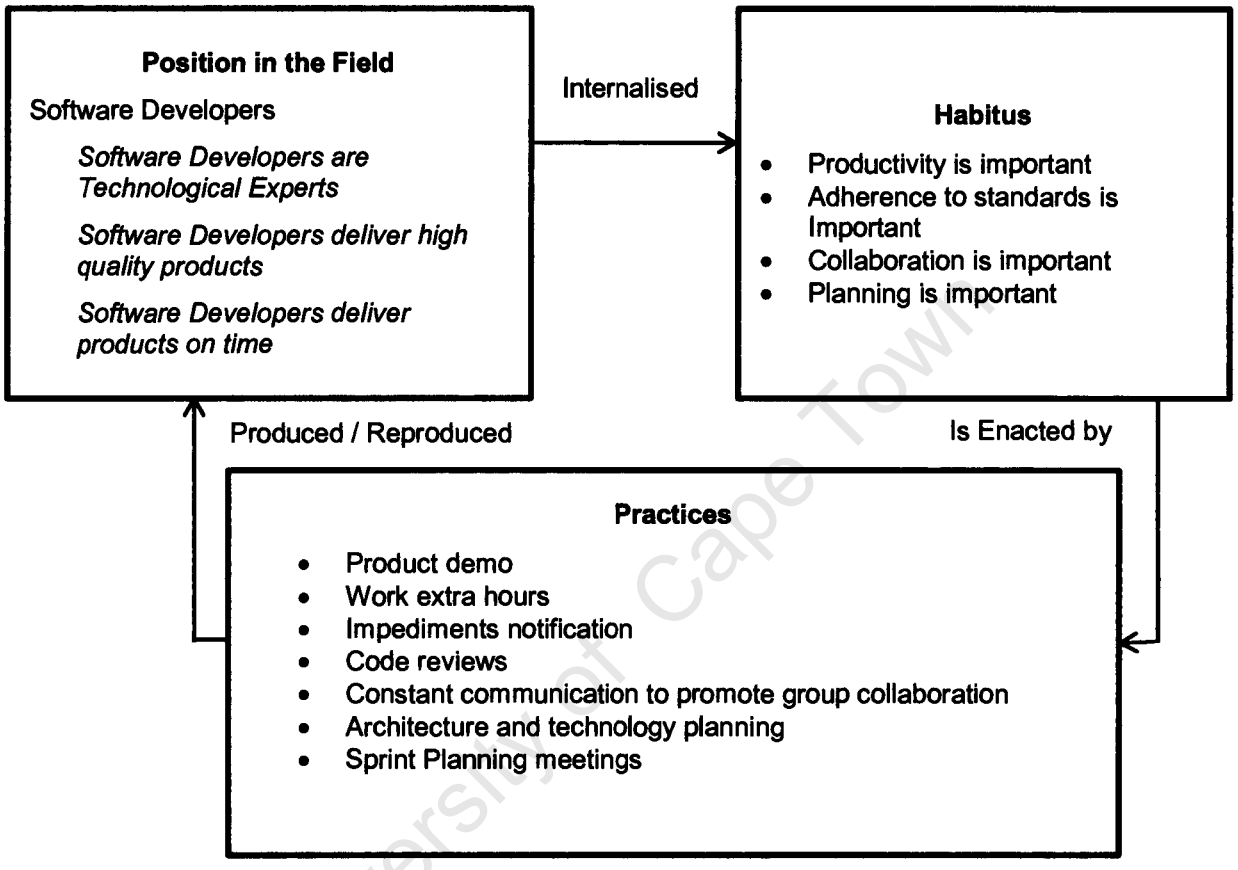


Figure 6-4 – Software Developers' Work Practices Circuit of Reproduction

It could be seen that agents in the position *Software Developers* in the *C2 GASD Team* field, were in positions of subordination or equivalence, in comparison to other agents in that field. However, their respective positions of power in the *Pune Team* field differed whereby some of them were in dominant positions in that field. This is because the *Pune Team* field was characterised by its own set of cultural and symbolic capital, of which the software developers possessed a substantial amount.

Relevant forms of symbolic capital pertained to the agents' degree of technical knowledge of software development tasks for the Colin application, as well as their academic qualifications. The relevant forms of cultural capital related to their seniority level. No form of economic capital was found relevant to that field. The two agents in

dominant positions in the *Pune Team* field were C2.R3 and C2.R6. C2.R4 was in an equivalent position, while C2.R5 and C2.R7 were in subordinate positions. The agents' cultural and symbolic capital has been described in Tables 6.4 and 6.5 respectively.

Agent	Degree of technical knowledge of the software development tasks for Colin Application	Academic Qualification
C2.R3	Highly knowledgeable about the Colin application, as has been involved in the project since its beginning	Bachelor of Computer Science
C2.R4	High architecural and infrastructure knowledge about the Colin application because of his previous onsite coordinator role	Master in Computer Science Engineering
C2.R5	Limited knowledge of the Colin application	Bachelor in Computer Science
C2.R6	Expert in the Colin application project because of his involvement since its beginning, both in South Africa and in India	Master in Computer Application
C2.R7	Limited knowledge of the Colin application	Master of Computer Science

Table 6-4 - Cultural capital for the Pune Team field

Software developers in the Pune Team were perceived to be technological experts. They took pride in their degree of expertise on the Colin application and ensured that their status as "experts" was maintained. They were also known to deliver high quality products, on time.

Agent	Seniority Level
C2.R3	Senior
C2.R4	Not Senior
C2.R5	Junior
C2.R6	Senior
C2.R7	Senior

Table 6-5 - Symbolic Capital for Pune Team field

6.3.1.2. Habitus and Practices of Software Developers

The analysis identified four forms of habitus which served as the generative basis for practice in the *Pune Team* field. These relate to the importance of productivity, the need to follow standards, collaboration and planning. Each form of habitus is discussed below along with the corresponding sets of practices.

- *Productivity is important*

Software Developers valued the need to be highly productive in the *Pune Team* field. Throughout the sprint, it could be observed that the *Software Developers* always strived to complete all the tasks which had been allocated to them, even during instances where they overestimated the amount of work which could be feasible in one sprint, as can be seen in that extract:

> *Our goal as a team, whatever the problem is at the beginning of a sprint, is to deliver that part. If we really overestimated a large amount, then we try to negotiate with the customer. If not, if it is a small percentage, then we try to do it* (C2.R3).

C2.R6 even admitted that the *Software Developers* in the team had a "doers" mentality: *This is a kind of "doers" team. So most of the time, it has to be done. That's the spirit in the team* (C2.R6). The *Software Developers* systematically strived to, as much as possible, complete any work requested from them. *Software developers* strived to maintain their productivity level by being predisposed to engage in various forms of work practices which are discussed.

- ***Product demo***

Given their habitus around the importance of productivity, *Software Developers* were predisposed to recognise the importance of having product demo as part of the Scrum process. They used the product demo as an opportunity to showcase their achievements throughout the sprint, thus obtaining formal recognition from the customers of the fact that they had completed what was expected of them, as shown from this extract:

> *At the end of the sprint, we have this demo with the product manager and other stakeholders where they look at the features [...] It is some kind of high level acceptance on whatever has been developed* (C2.R3).

In cases where defects or opportunities for improvements were uncovered, these were further specified as user stories for the upcoming sprint. However, whenever possible, they tried to fix the urgent defects immediately: *And during the demo, there are sometimes defects which are taken as features for the next sprint. Some defects are fixed immediately* (C2.R3).

Through these work practices, the *Software Developers* defended their positions in the field pertaining to always delivering high quality products. The product demo was particularly important to obtain formal recognition from the customer that whatever was being developed was approved and could not be flagged as being incorrect later in the project life-cycle, which would jeopardize their position.

- ***Working extra hours***

Software Developers also predisposed to work extra hours given their habitus. For example, at the time of the study, the team had missed their project deadline and, given the fact that the *stakeholders were very serious about delay* (C2.R4), they were all expected to work during weekends and after normal work hours during the week, as shown from this extract: *That is why during the past 1.5 months, we were coming on weekends and we were staying late* (C2.R4).

The *Software Developers* did not feel that working extra time was a burden to them. They felt that by working extra hours, they were enhancing their knowledge and asserting their commitment to Sanbi and the customer. However, the reason behind this work practice was that they were striving to maintain their productivity level, and consequently reinforced their position in the structure of the *Pune Team* field:

> *If the work is there we will come during weekends, but we do not feel that we are working for the company. It is more as if we are acquiring knowledge. Whatever the company is giving to us, we also give back to the company (C2.R4)*

> *I think how we are looking at it is that we have to complete this at this day, and whatever the customers are asking we have to do that (C2.R3)*

> *If something goes wrong and we have to make up for that in terms of deadline, then we have to spend extra hours (C2.R6)*

- ***Impediments notification***

Software Developers in the *Pune Team* field reported on any impediments that they experienced during the sprint whenever they occurred. Impediments often related to specific issues which prevented them from completing the tasks assigned to them on time (e.g. lack of information on a specific user story, technological issues etc.). Instead of waiting for the daily Scrum meetings where such impediments are usually reported, they immediately informed the customers onshore of the situation.

> *Suppose that I've given an estimate of two days for my task. And after about half a day, I come to know that it is technically more difficult and I will require one more day, so I don't wait for the Scrum meeting on the next day. I talk to Casper over Skype and inform him about the problems that I am facing. If possible, he will contact the technical person there and try to get some resolution for me (C2.R3).*

Whilst reporting impediments, the *Software Developers* not only sought to inform everyone that their work was on hold, but also attempted to obtain quick resolutions to their problems. They employed various strategies to ensure that they obtained the resolution quickly. For example, C2.R4 explained that when he failed to obtain the required help from someone onshore, he would instead contact someone higher in the work hierarchy who would then "pressurize" the latter to collaborate. Informing all parties about impediments also exonerated them in case of delays:

> *If the product manager is however very busy with some other projects, his responses might be slow. The project manager is also occupied with some other projects but his responses are more in time. So if the product manager is very busy and if we want to get our job done quickly, we communicate with project manager who can in turn push the product manager to give more timely answers* (C2.R4).

These practices enabled the software developers to deliver their work on time and thus maintain their position in the field. They also defended that position by ensuring that they would be exonerated in situations where their work was delayed due to external circumstances.

- *Adherence to standards is important*

The *Software Developers* also valued the adherence to standards prescribed by the agile community. They expressed the need to adhere to the agile precepts rigorously and to

learn about these principles either through formal training or from other colleagues who they thought were highly skilled on the topic of agile software development, as shown from this extract:

> *I had very good lessons from one of my team leaders in my previous company. From the very beginning, I was doing the right thing when it comes agile. I wanted it to work. I wanted to be an agile developer when I learnt about it. Obviously, initially I did not understand everything about agile, but I found some things interesting just by instinct. And I started following it rigorously* (C2.R6)].

At the same time, they also believed in the flexibility of Scrum as a methodology. They felt that some aspects of the methodology could be adapted based on the project conditions and requirements of the customer. For example, they wrote documentation as required by the customer and ensured (to some extent) that the documentation was kept up to date:

> *Again it depends on the project on a case-by-case basis, but I wouldn't really put it as a disadvantage of agile or a disadvantage of scrum. Scrum never says that we cannot update the design. We can still follow the theoretical scrum, and still have design documents up to date* (C2.R3).

Software Developers also valued the quality and standard of the code which was being written. They felt that they always tried their best to maintain the quality of the product and the code standard for the task which had been assigned to them. As a team, they all shared the responsibility for maintaining a high code standard, as shown from this extract:

> *Sometimes it happens that a customer complains about our code quality to Project Managers, so in that case we know that we must focus on quality. So doing the best is one kind of motivation that we have among all the team members* (C2.R4).

The practices which the *Software Developers* were predisposed to engage in, given this habitus, are described below.

- ***Code reviews***

Given their habitus on the need to follow standards, *Software Developers* were predisposed to engage in code reviews. Such a practice, put forward by the customers to ensure the quality of Colin application, was perceived as normal and even necessary by the software developers who valued the adherence to standards within the team. In spite of the distance, they managed to implement a code review process through the use of Skype, as shown in this extract:

> *[The code reviewer onshore] opens the same code on his machine. You tell him that you've developed class X. He is a technical guy so he asks us about what happened and why we did such and such in the class. Then we explain the thought behind the methods and logic in the code. Skype is just to talk and we try to optimise the code and follow good coding standard* (C2.R3).

To facilitate the reviewing process and maintain good coding practice, the *Software Developers* also inserted comments in their methods: *We usually comment the codes that we have done and the technical person can review our codes if he wants to (C2.R5)*

The practice of having code reviews reinforced the *Software Developers'* position in the structure as they could demonstrate the quality of the product they were implementing. In cases where the code standard was not good, they could learn on how to improve their algorithms and leverage their technical expertise and thus acquire more cultural capital, in turn strengthening their position in the field.

- *Collaboration is important*

The *Software Developers* had previous experience in GSD and understood the need for effective collaboration. They had thus applied this belief in their daily work on the Colin application. Specifically, the appreciation of collaboration across dispersed teams was transposed from the GSD community to this nested field.

The *Software Developers* not only valued collaboration across the sites but also amongst themselves on a daily basis: *We do tend to work in groups sometimes, I'm not sure if it applies to Indian people all across. But from what I've seen there is a tendency to work collaboratively (C2.R4).* The practices which the software developers were predisposed to engage in given this habitus are described below.

- ***Constant communication to promote group collaboration***

Given their habitus on the need to collaborate across and within the two sites, *Software Developers* were predisposed to engage in various work practices around communication, to further enhance their subsequent collaboration. For example, they often engaged in discussions on task estimation prior to and during the sprint planning meetings. Upon receiving the list of user stories which the Product Owner and the Scrum Master wished to include in the sprint, an initial meeting was held onshore amongst the software developers, where they estimated the complexity and duration of each task for the given user stories:

> *We are given user stories and 14 features are there. So we discuss among ourselves and we talk to our manager to say that this is our estimation* (C2.R4).

Collaboration across the sites was also achieved when all the stakeholders (onshore and offshore) then discussed each task over the phone to finalise its estimated duration:

> *The people present are product manager, project manager, team lead and all the developers and QA. It happens over the phone. And we discuss the complexity of each task and we finalise the user stories we will work on. And after that we get final confirmation about our analysis within the next day. Because we do our analysis and then we raise some queries to the product manager and he finalises those queries and then we start* (C2.R4).

In addition, *Software Developers* also discussed amongst themselves the best way to complete a task in case of uncertainty. For example, if they did not have enough knowledge about a certain module while their colleague did, they often collaborated with that expert to identify how best to design the code:

> *Suppose that the feature is very simple and I know the dependencies of the modules that I will be developing, then I go straight forward. But if I know that the feature might have some dependencies with some other module that I'm not familiar with, then I discuss with my colleagues and I get the feedback from him on how to do the design (C2.R6).*

The communication channel between the onshore and offshore teams was also open, and

Software Developers also collaborated with experts onshore in cases of emergencies:

> *It's very easy and they [onshore team members] are very approachable. Whenever you are in difficulty you can directly call them. You can ping a message, saying that this is a requirement that I am a little bit confused with, please clear it. So it's very straight forward* (C2.R6).

While engaging in practices to enhance and maintain group collaboration across and within the sites, software developers maintained their positions in the field, as collaboration enabled them to deliver high quality software on time.

- *Planning is important*

Similar to the habitus pertaining to the need to collaborate, the need to plan was transposed from the GSD Community field. In addition, given the high importance of planning derived from the Scrum community, the software developers further valued its importance. The various work practices to which the *Software Developers* in the Pune Team field were predisposed, given the planning habitus, is described in the following sub-section.

- ***Architecture and technology planning***

Given their habitus around planning, *Software Developers* were predisposed firstly to pay attention to the architecture and adequacy of the technology employed while embarking on a new software development project, as shown in this extract:

> *Suppose that I am starting with some new project, I must investigate which technology I will be using? If we have suggested a Model View Controller (MVC) for the architecture, we need to do a study of that thing to see whether it is suitable for the project or not, and we also do proof-of-concept for some critical things* (C2.R6).

Such research tasks were included in the initial sprints prior to any software development work and were thus recognized as essential stages during the overall software development process:

> *These could be conducted at the sprint level. For example, in sprint 1 we will do only proof of concept, whether this new technology will suite our project or not. But that*

should not be more than sprint 1 and 2. And then the development part can start and that is covered in each sprint (C2.R6).

- ***Sprint planning meetings***

The sprint planning meetings were divided into two parts within the C2 GASD team: "sprint planning 1" and "sprint planning 2". During the first sprint planning meeting, the *Software Developers* strived to obtain as much information as possible on the user stories scheduled for the current sprint. They first listened to the explanations provided by the Scrum Master and the Product Owner and were then given the opportunity to ask for further clarifications. This meeting was sometimes held via video conference if the connectivity was good, or otherwise via phone conference calls:

Time is given separately for analysis and design. For example, sprint planning is generally video conference and Casper and Philip on their side and the entire team here sit together during a video conference. And they explain what are the backlog items they are thinking of introducing for this sprint. For each item, they try to explain as much as possible the scope of that feature, and what the end user really wants. At that time we don't discuss anything technical. And after that session, time is given for raising out queries. Maybe after doing some feasibility study, we may come across some questions or queries (C2.R3).

The user stories were assigned to specific developers at the end of the "sprint planning meeting 1". Often, experienced software developers were allowed to choose the user stories they wished to work on, provided that these were in line with their area of expertise. *Software Developers* with limited experienced were not allocated complex tasks in order to maintain the quality of the codes being written:

For example Naveen is new and has never worked on calculation. If we give that feature to him, he will take more time. Quality will be lowered or I will end up always going at his desk to explain how to do that. So it will waste my time as well as his time. So it is better to allocate features based on expertise (C2.R4).

Software Developers took care not to over-commit themselves to the number of stories which they could complete in one sprint and organised themselves so that the user stories were completed simultaneously whenever possible: *It's done in parallel. One feature is given to a maximum of two people. If there is dependency between two*

features then we do it sequentially. Otherwise things will not be completed in four weeks (C2.R3). In addition, they planned for user stories of high priority to be completed first.

During the second sprint planning meeting, the discussion was centered on the technical aspect of the user stories. In particular, the *Software Developers* focused on breaking down each user story into specific tasks:

> *The next day, we have another Video Conference (VC) to discuss the technical things. Based on our knowledge on the first VC, we create a design document and we upload that to the wiki. Each developer is given one or two features each and we will discuss that feature over VC. And then we try to explain our technical approach to handle that feature. We say what are the queries and what are the bottle necks. And after the second VC, the scope is finalised. Ideally there should not be any change in that feature* (C2.R3).

6.4. The Durban Team Field

The team members at C2, based in Durban, were seen to belong to a separate nested field labelled as *Durban Team.* The *Durban Team* field had two positions namely: *Customers* and *Onsite Coordinators.* The position, habitus, and work practices of the *Customers* and *Onsite Coordinators* are described in Section 6.4.1 and 6.4.2 respectively.

6.4.1. Customers' Work Practices

To shed light on the work practices of *Customers* in the *Durban Team* field, each element in Figure 6.5 is described.

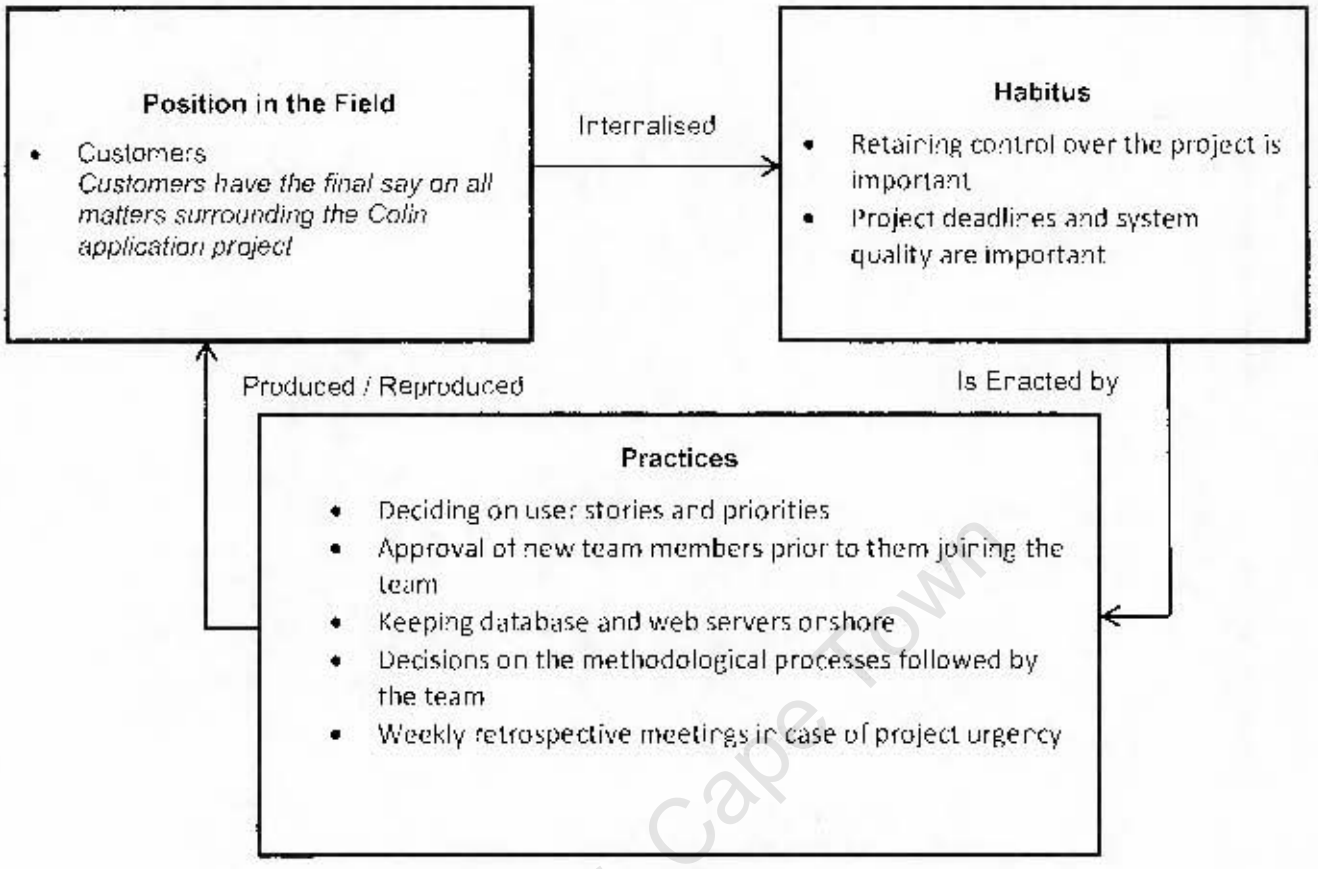


Figure 6-5 – Customer's Work Practices

6.4.1.1. Customer's Positions in the Field

In the *Durban Team* field, *Customers* were in a dominant position of power. Their position in that field was such that they *had the final say on all matters surrounding the Colin application project*. Two agents (Casper and Philip) held the customer position in the field. Their dominant position in the field was due to the specific form of symbolic and cultural capital which they held in the *Durban Team* field.

Agent	Degree of expertise in the Business Logic of the Colin Application
Casper	Expert and highly knowledgeable in the Colin application business logic and excellent knowledge on the types of requirement needed for the Colin application to bring business value
Philip	In his capacity of Product Owner, expert and excellent knowledge of the business logic of the Colin application
Yogi	Low degree of expertise and limited knowledge of the business logic of the Colin Application.

Table 6-6 - Cultural Capital for the Cape Town Team field

In terms of their cultural capital, *Customers* had a high degree of expertise on the business logic of the Colin application (see Table 6.6). In addition, given their job titles of Scrum Master and Product Owner respectively, agents in the *Customers* position also held a high amount of symbolic capital which also contributed to their dominant position (see Table 6.7). Similarly to the *C2 GASD Team* field, economic capital pertaining to the degree of control on the Colin application project was also relevant to the *Durban Team* field and both agents in the customer position had high amount of economic capital.

Agent	Degree of ownership on the Casper Application Project
Casper	High degree of ownership because of his ability to elicit and manage the requirements for the Colin application on behalf of the Customer's organization.
Philip	Some degree of ownership as he is the official Product Owner for the Colin application project
Yogi	No ownership of the Colin application project.

Table 6-7 - Economic Capital for the Cape Town Team field

Agent	Job Title
Casper	Scrum Master / Project Manager
Philip	Product Owner
Yogi	Onsite Coordinator

Table 6-8 - Symbolic Capital for the Cape Town Team field

6.4.1.2. Habitus and Practices of Customers

The study found that the *Customers* from the *Durban Team* sub-field had internalised two forms of habitus: *retaining control over the project is important, project deadlines and quality are important*. These three forms of habitus are further described below, along with the sets of practices which the customers are predisposed to engage in.

- *Retaining control over the project is important*

Based on the discussions involving the customers and top management at Sanbi prior to the beginning of the project, it was felt that the customers onshore highly valued the need to coordinate and control any issues around the Colin application project. The need for control was attributed to a lack of *faith* (C2.R9) in the Indian team members. Consequently, top management at Sanbi explained that Scrum was chosen as opposed to

XP for the Colin application project as it was perceived in the field that it would enable them (the customer) to better manage the resources around the project. In contrast, they felt that XP empowered the software developers, and they wanted to avoid this situation:

> *Why did we choose scrum and why we didn't go for XP? The basic thing was the management principle. If somebody is going for Scrum, management is more responsible than resources. But in XP, the management had more faith in the people driving that. That's why we went for scrum* (C2.R9).

Even the software developers felt that their onshore customers wanted complete clarity on the daily work progress: *They want minute-to-minute clarity. They don't want to give control to the offshore team* (C2.R3). The set of practices which the *Customers* were predisposed to engage in, given this habitus, are described below.

- ***Deciding on user stories and priorities***

Given their habitus, whereby the *Customers* value the need to retain control over the project, the latter are predisposed to decide on both the user stories' content and their respective priorities. In practice, Casper and Philip communicated with the online casino owners (i.e. the end users) to obtain their views on what the new requirements for the software should be:

> *They get the requirements from the operators directly. [The customers] have direct contact with the operators. They have 5 to 6 operators in Europe. Operators are casino owners. They have online casino owners which run multiple casinos. These are not physical casinos, they have websites. Players come online and lose money and the casino owners earn money* (C2.R3).

They also obtained new requirements by *comparing the software with other similar products to determine what new features can be introduced* (C2.R3). These requirements were then written in the form of user stories.

In addition, the *Customers* also decided on the respective priorities for each user requirement. However, their decisions were at times influenced by other stakeholders (e.g. company owners) in Durban: *Pertaining to priorities, that part stays more with*

Casper and Philip I think. But then again, the stakeholders push them to increase the priorities. It's not always the ideal world that you live in.

- ***Approval of new team members prior to them joining the team***

Customers retained their position in the *Durban Team* field by putting forward a rule that any new member introduced to the software development team offshore would need to be approved by them. They required that any new team member be temporarily involved in software development work for the team for one iteration, after which they would assess his or her performance.

- ***Keeping database and web servers onshore***

Customers also retained control over the project by keeping all databases and web servers onshore. Software developers in India did not have direct access to the test and live data. In the event of a web server being down, the software development work was often delayed as it was not always possible to reboot the server immediately. The *Customers* were aware of the negative impact of delays on the software developers' ability to meet the project deadlines, but they nevertheless refused to set up a testing environment with the adequate servers and test database offshore. They justified their decisions by the need to secure their data.

> *At the moment, these people [software developers] have their code sitting on their machines. But the database is located in the client environment. Then their web services and webservers are still in the client environment. So we have discussed whether we can have this environment here [offshore] for testing purposes. But for security reasons, they [customers] don't allow it* (C2.R8).

While data security was a valid reason for keeping the database onshore, this practice also allowed the customers to control the process and retain their dominant position in the field whereby they still decided on all matters surrounding the project.

- ***Decisions on the methodological processes followed by the team***

Customers also maintained their position in the field by enforcing specific decisions on how the Scrum process should be followed by the team. For instance, the burn-down chart, and task board were managed onshore. They would gather information about progress being made during the sprint during the daily stand-up meeting and update their task board and burn-down chart accordingly. Pictures of these artifacts would then

be sent offshore. *They have a scrum board there with the post-it notes. They used to send pictures of those. They update the burn-down chart onshore* (C2.R8). In addition, while all software developers were responsible for updating their personal progress for their allocated task on the wiki, the *Customers* ensured that they were the one responsible for creating new tasks in the wiki at the beginning of the sprint: *Everybody has responsibility to update the wiki. But for the creation of the task, it is Casper* (C2.R5).

When project tasks completion was critical and the project delays were encountered, the customers also bypassed the update of the Scrum artifacts. They instead requested that the software developers dedicate all their time to completing the work at hand. For example, from sprint seven onwards, numerous bugs were identified which had to be fixed urgently. Consequently, the update of the burn-down chart and the task board was no longer undertaken and progress tracking via Scrum artifacts was bypassed: *From sprint 7 onward, it is a mix of feature plus bug fixing. And it is difficult to track the bug fixing and the project estimates, and there are so many. So it was quite difficult to track and we stopped doing it from sprint 7 onwards* [C2.R8].

- *Project deadlines and quality are important*

Customers in the *Durban Team* field also valued project deadlines and quality. In particular, software developers felt that the customers expected them to complete whatever the latter requested: *I think that they believe that whatever they ask for, we have to do it* (C2.R8). The software developers also explained that no one was allowed to miss a deadline and that the project plans put forward by the customers had to be followed diligently: *If the project manager has laid down some project plans, if something is making us miss a deadline, that won't be entertained. We are not allowed to miss a deadline* (C2.R6).

In addition to project deadlines, the *Customers* in the *Durban Team* field also valued system quality. This can be perceived in some of the practices which they had put in place in the C2 GASD Team and which have been described in the previous sections, namely: Code Reviews and Product Demo. As mentioned by C2.R3: *the customers are very much concerned by quality*. The practices which the customers in the *Durban Team* field were predisposed to follow, given this habitus, are described below.

- ***Weekly retrospective meetings in case of project urgency***

Given their habitus valuing the need to meet project deadlines and maintain quality, *Customers* were predisposed to implement work practices where they would be able to constantly monitor the work progress of the software developers. For example, when tasks had to be completed urgently or when the software developers had missed some deadlines, the *Customers* requested weekly retrospective meetings, to be held with the project manager offshore. Under normal circumstances, these meetings were held monthly. Weekly retrospective was used by the customers as a means to closely monitor the progress being made and to implement any measures to ensure that the software developers are more productive:

> *We have weekly review meetings or monthly review meetings that are happening with the customer. So there we are tracking whether we are going in the right direction or not. Currently, just because of time constraints, it is weekly. Cause we have to finish things on time and we do weekly* (C2.R8).

6.4.2. Onsite Coordinators' Work Practices

To shed light on the work practices of *Onsite Coordinators* in the *Durban Team* field, each element in Figure 6.6 is described below.

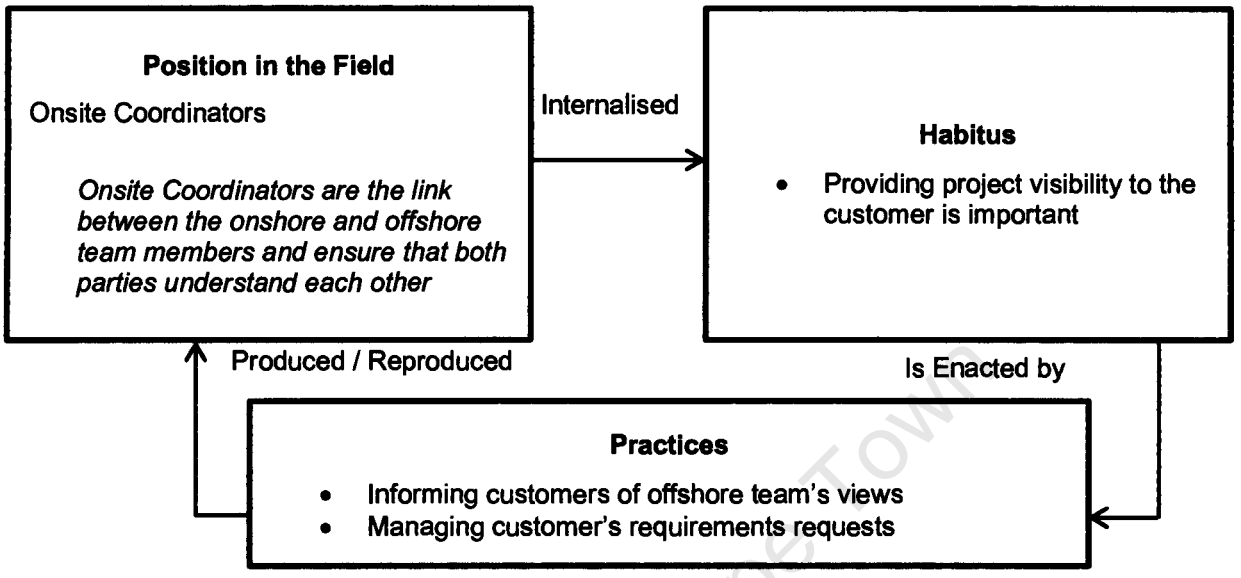


Figure 6-6 – Onsite Coordinators' Work Practices

6.4.2.1. Onsite Coordinators Positions in the Field

Onsite Coordinators were in an equivalent position in the *Durban Team* field. In terms of their position in the field, onsite coordinators were perceived as the link between the offshore and the onshore team members. Their role was thus to ensure that there were no misunderstandings between the two parties, given the physical distance between them. Only one agent belonged to that position (Yogi).

Onsite Coordinators held an equivalence position in the *Durban Team* field, given the amount of cultural, symbolic, and economic capital they owned. The relevant form of cultural capital for the *Durban Team* field related to the degree of expertise of an agent in the business logic of the Colin application. Even though the *Onsite Coordinator* was a technological expert, his knowledge of the business logic of the Colin application was limited in comparison to the other agents in that field. He thus held a limited amount of cultural capital in the field. The amount of economic capital held by the *Onsite Coordinator* was also limited as he had no control for the Colin application. However, given his job title, which was recognised as a key role within the team, he held a sufficient amount of symbolic capital which enabled him to be in an equivalent position.

6.4.2.2. Habitus and Practices of Onsite Coordinators

Only one form of habitus was found relevant to the *Onsite Coordinators* work practices circuit of reproduction in the *Durban Team* field, namely: *Providing visibility to the customers is important*. This habitus is further described below.

- *Providing project visibility to the customers is important*

The role of *Onsite Coordinators* was primarily created because it was felt that the customers onshore wanted to have a constant visibility on what the software developers offshore were working on, their work progress and the amount of time spent daily on particular tasks. Given the fact that the daily stand-up meetings were not enough to provide the required degree of visibility, this specific role was created. Consequently, the main purpose of onsite coordinators was to introduce this form of visibility and this value was inculcated to agents belonging to that role because it had been collectively orchestrated:

> *The client indicated that they not have a clear idea of what was happening offshore, even if quality was good and we were delivering in time. They wanted to know 8 hours a day, which resource was working on what task. This was not visible to them. We were discussing how to send the status and in which format to send the status to them so that there is no misunderstanding. And we found that just the daily scrum was not enough for that. So we introduced a coordinator there* (C2.R8).

The practices which the onsite coordinators in the *Durban Team* field were predisposed to follow are described below.

- ***Informing Customers of Onshore Team's Views***

Given the habitus of providing project visibility to the customers, *Onsite Coordinators* were predisposed to ensure that the views of the offshore team were always presented to the customers. Consequently, in addition to providing work progress feedback, *Onsite Coordinators* also strived to inform the customers about any impediments which the offshore team faced, causing them to remain unproductive for some part of the day. The *Onsite Coordinators* thus acted as an advocate for the offshore team members, and maneuvered to resolve any impediments which they faced by liaising with the customers onshore:

If we are facing connectivity issues, he has to coordinate there. If we are facing connectivity issues, he has to coordinate there (C2.R5)

Suppose [the customers] were not clear about why I was sitting idle for the first part of the day and they were suspicious, it is my responsibility to clearly detail my schedule to the onsite coordinator. And he will easily communicate that to the customer (C2.R3)

- ***Managing customer's requirements requests***

Prior to the introduction of *Onsite Coordinators* in the team, customers would put forward user requirements to the offshore team which the software developers felt were not technically feasible or clear. The role of the *Onsite Coordinators* was thus to intercept these requirements, assess their technical feasibility and clarify them before passing them on to the software developers offshore: *He should know what exactly the problems are, what exactly the client means and whether that feature is technically feasible or not* (C2.R3). By ensuring that only clear and precise requirements were passed on to the offshore team, the productivity of the software developers was maintained as no time was wasted in to and fro conversation between the onshore and offshore team to obtain clarifications.

6.5. The Sanbi (C2 Organsiation) Field

Following analysis, it was found that an important position in the Sanbi (C2 Organisation) field is the *Project Managers* position, whose habitus and work practices influences the C2 GASD Team field circuit of reproduction. Consequently, a description of the work practices of agents in the Project Managers positions in the *Sanbi (C2 Organisation)* field will be provided. Other positions are also relevant to the *Sanbi (C2 Organisation)* field, but have not been described as they were not found relevant to the research question.

6.5.1. Project Managers' Work Practices

To shed light on the work practices of *Project Managers* in the *Sanbi (C2 Organisation)* field, each element in Figure 6.7 is described below.

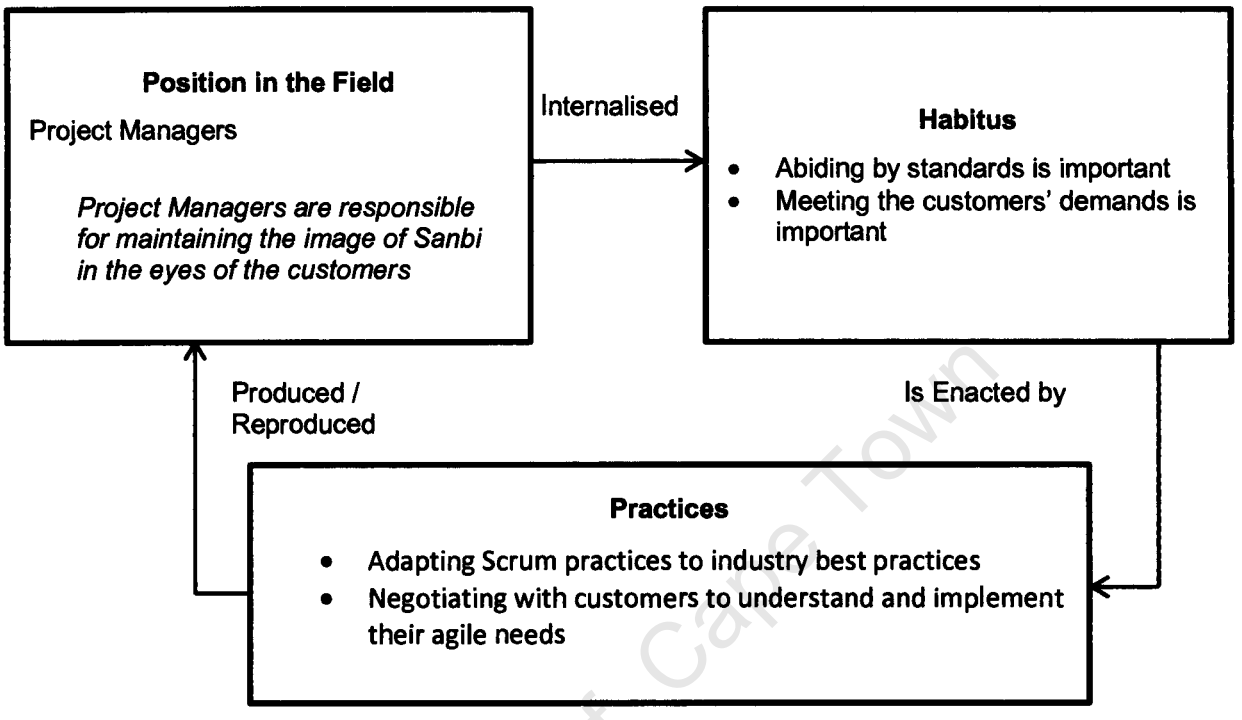


Figure 6-7 – Management's Work Practices

6.5.1.1. Project Managers' Positions in the Field

As can be seen in Figure 6.7, the *Project Managers* position in the *Sanbi (C2 Organisation)* field was such that they were perceived as the ones responsible for crafting and maintaining the image of Sanbi as an organization, to the eyes of the current and potential customers. Agents in the *Project Managers* position were in a dominant position and were able to make important decisions which impacted the entire organisation. They often implemented practices (e.g. the implementation of Scrum teams for specific customers) driven by a vision of where the organisation, as a whole, should be positioned worldwide. They also strived to maintain and increase the competitiveness of Sanbi worldwide.

6.5.1.2. Habitus and Practices of Project Managers

Two forms of habitus were found relevant to the *Project Managers* position: *abiding by standards is important,* and *meeting the customers' demands is important.* The habitus are described below, along with their sets of practices.

- *Abiding by standards is important*

The analysis noted that the agents belonging to the *Project Managers* position valued highly the need to abide by various forms of standards in order to maintain their credibility in the eyes of the customers. For instance, upon deciding to adopt agile methodologies in some of the delivery units at Sanbi, they felt that it was important not to blindly follow any *myths* around agile software development and that an accurate theoretical understanding was rather required. Formal training was then undertaken to ensure that they abided by the correct agile principles and Scrum work practices:

> *The only challenge is the way in which the agile term is used. When we are using this term, we need to be quite clear on what we mean because there are some myths in agile as well. Some people tend to think that there are frequent releases during the development phase then it's agile. So people sometimes have these kinds of myths. But then they get the formal training its fine* (C2.R9).

In addition, *Project Managers* felt that they had to abide by the CMMI standard. They believed that once that they had defined the software development practices which would be followed at Sanbi, and aligned them with agile work practices, it would not be difficult for them to adhere to CMMI practices:

> *Actually, CMMI allows you to have agile processes. Once you've defined what you are going to do and if you can follow that it's fine. It's at a very high level of CMMI as well that you define what you are going to do. Of course you have to follow some principles which are prescribed by them. And then if you follow that, you are OK* (C2.R9).

They also valued the need to implement design and coding best practices, irrespective of the methodology being used. This habitus was transposed to the *Pune Team* sub-field:

> *At the technical level, we have the design best practices, coding best practices. Those are still applicable irrespective of whether we are using agile or waterfall. Design principles may slightly change. But the coding guidelines and best practices will remain the same* (C2.R9).

Project Managers also followed the trends set forth by experts and leaders in the IT industry, like Gartner. They believed that in doing so, they would diversify their service offerings and maintain their competitiveness in the market.

- ***Adapting Scrum work practices to industry best practices***

Project Managers adapted Scrum work practices to the industry best practices in various ways. As previously mentioned, they attempted to merge agile and CMMI practices to ensure that the two did not conflict each other, as both standards were valued in the organization.

As already mentioned they followed industry experts and trends established by them on IT best practices and attended agile training courses, in order to obtain the correct knowledge. However, the agile trainers were selected with care and they also clearly specified the type of knowledge they wanted their employees to acquire, prior to the training: *We made use of outside consultants in this, and we made sure that they really taught us what we wanted. And the instructors are normally certified scrum masters and scrum practitioners so we were assured to get the correct knowledge* (C2.R9).

- *Meeting the customers' demands is important*

Agents in the *Project Managers* position valued the need to be in tune with the demands and needs of their customers, in order to provide the right service to them. This habitus was transposed to the *C2 GASD Team* sub-field. For example, they were aware that some of their customer pools wanted faster product delivery to leverage their competitiveness and profitability. *Project Managers* thus ensured that the right practices were implemented to satisfy these customer demands:

> *The customer needs depend on that. They want faster time to market. They want their product to go to the market. Let's say I'm a product organisation and I'm creating a product. Some other organisation has a similar product and they are having more advanced features, I need to adapt to those. So the competition is very high and then we turn to agile. They fit together very well* (C2.R9).

- ***Negotiating with customers to understand and implement their agile needs***

Project Managers strived to understand the demands of the customers, either based on direct requests from them, or from conversations geared towards uncovering their exact needs and proposing potential solutions to them:

Suppose the customer wants to go from the traditional to the agile. Then we can dialogue with them to get some form of understanding. Or in some cases, they might have implemented certain things in other projects, using agile. So it's a kind of give-and-take to come to a common sort of understanding with them and how would we go forward in such case. But as long as both us and the customer agree about the basic principles of Scrum or XP then we have less problems (C2.R8).

6.6. Chapter Summary

This chapter described the various overlapping, joint and nested fields identified for the C2 case study as well as their relationships with each other. In particular, four key fields were found relevant to C2, namely the: *C2 GASD Team* field, *Pune Team* field, *Durban Team* field, and *Sanbi (C2 Organisation)* field. The circuits of reproduction of the work practices prevailing in these fields have been described, paying particular attention to the positions, habitus, practices and forms of cultural, symbolic and economic capital. Information provided from this chapter and Chapter 5 will be used as a basis to describe and explain the Scrum process breakdowns identified in the two cases in Chapter seven.

7. SCRUM PROCESS BREAKDOWNS EMPIRICAL OBSERVATIONS

This chapter describes the Scrum process breakdowns identified from the empirical analysis of the data from the two aforementioned cases. The Scrum process breakdowns described in this chapter are to be understood given the contextual description of the practices detailed in Chapters 5 and 6. The chapter also elaborates on the social conditions under which these breakdowns were seen to occur, in light of Bourdieu's Theory of Practice. The chapter is divided into three sections. The first section focuses on the breakdowns occurring during and after the sprint planning meetings. The second section focuses on the breakdowns occurring during and after the retrospective meetings. The third section compares the Scrum process breakdowns identified in the study to what was specified in the literature, to highlight similarities and differences. A theoretical discussion on the social conditions leading to the Scrum process breakdowns is then proposed in Chapter Eight.

7.1. Scrum Process Breakdowns Occurring During and After Sprint Planning Meetings

The Scrum process breakdowns occurring during and after sprint planning meetings could be categorised as sprint interruptions and collaboration breakdowns. Sprint interruptions pertained to *impromptu changes to user stories content and priorities.* Collaboration breakdowns pertained to *Product Owner's low level of authority, different perceptions about task urgency between the software development sites, disagreements on software engineering practices, number of user stories to be completed in the sprint being imposed on the team, low level of communication openness,* and *disagreements on estimation mechanisms.* Appendix 29 summarises the Scrum process breakdowns identified during and after the sprint planning meetings and the conditions and factors leading to them in the case studies.

7.1.1. Sprint Interruptions: Impromptu Changes to User Stories Content and Priorities

The interruptions during the sprint indicate that the agreements reached during the sprint planning meeting did not hold throughout the sprint, thus giving rise to breakdowns after the sprint planning meetings. The sprint interruption identified in both cases was the impromptu changes in user stories' content and priorities in the middle of

the sprint. The user stories and their respective priorities were agreed upon following negotiations between the team members and the Product Owner, during sprint planning meetings but sometimes changed mid-sprint. These changes could occur at any point in the middle of the sprint, and the later it occurred in the sprint, the more disruptive they were to the team. At C1, the impromptu changes, in content and priorities, in the user stories were always put forward by the CEO. The Product Owner was aware of the problem as indicated by his statement:

> *"What happened sometimes is that the CEO is very much involved with the project. In the middle of a sprint, requirements might change and he will reset the priorities of the tasks or change the user stories. This is a problem because it impacts on the programmers' workload and all the other tasks" (C1.R2).*

Changes in user stories impacted the team members negatively and resulted in loss of development time and stress, as can be seen by the following extract:

> *"I had a story in this sprint, and I had to throw it all away yesterday because they changed everything in there. They actually do not change the main concept of the story, but the story still changes. And this time I had to do it all over again* (C1.R7).

In C2, the software development team was also impacted by the modification of user stories in the middle of a sprint. The software developers felt that the onshore customers were themselves not fully aware of their own expectations from the Colin application project as they had inherited the project from another company. The software developers thus explained the impromptu changes in requirements by a lack of clear vision on what should be the core purpose of the project. They also attributed these changes to the casino operators' lack of understanding of the requirements and of the feasibility of the requests, as shown by the following extract:

> *"That has been the real problem with this product. Because it was inherited, the whole team doesn't have an exact idea of what they need from this project. So there are some faults around the core idea of the project. So that's why there is a change in the user stories all the time. [The customer] will say: "this is what I require" then suddenly they go "no that might not be the best way to do that". So because customers are not always very clear about what they want, there is always a change in requirements* (C2.R6).

No provisions for possible sprint interruptions were made during the sprint planning meetings at C1, thus exacerbating the negative impact of this Scrum process breakdown on the team and the project.

In contrast, software developers in C2 tried to allow for some form of time buffer while estimating the duration of a task. They felt that as a team, they took care not to over-commit themselves during a sprint: *Because companies don't over commit things in India because they like to have some sort of buffer* (C2.R6). Consequently, even though the negative impact of the impromptu changes to the user stories was felt in the team, its effect was less intense.

- ***Conditions Leading to Impromptu Changes in User stories' Content and Priorities***

In both case studies, the user stories' content and priorities were changing mid-sprint when dominant team members competed for more cultural, symbolic, or economic capital, and strategized to maintain their dominant position in the structure of the field. In the case of C1, the CEO was well aware of the negative impact of such impromptu modifications on the team members and on the project but nevertheless requested for these changes. He justified his actions by claiming to be promoting the quality and degree of purpose of the MKX application and thus found it relevant to make any changes required to the user stories at any given time, as can be seen from this extract:

> *"If I see that things don't work and that the flow is really not acceptable on the application itself, I will change the priorities. It's just important for me, the quality and the degree of purpose that's embodied in this application"* (C1.R5).

However, the empirical analysis revealed other motives for the implementation of these changes. It was noted that these impromptu changes allowed the CEO to defend his position as *driver of the MKX application* in both the *C1 GASD Team* field, as well as in the larger *HIM field* by constantly acquiring more economic capital (control over the MKX application) in both fields. C1.R5 had a high amount of economic, symbolic and cultural capital in both fields and was thus highly influential. By changing the user stories mid-sprint, it allowed him to acquire more capital by strengthening his control of the MKX application. This further reaffirmed his dominant position and ability to drive the MKX product vision in the C1 GASD field. In the larger HIM field, it allowed him to defend his

dominant position as an *"innovation leader"* (C1.R1) who would always provide useful requirements to the end-users whenever required.

The HIM group was global and comprised several organisations working in silos and at times in competition with each other. C1.R5 acquired the position of innovation leader within the larger HIM group by attempting to create synergies amongst these organisations through the use of the MKX knowledge management application. He stated:

> *"We overlook at what these companies are doing and try to create synergies amongst them. If somebody developed a nice application, we see if we can use it for the rest of the group instead of re-inventing the wheel. We are very much into knowledge management. We've created a knowledge management tool and this is what we are working on. The tool is trying to get everybody from the different companies to talk to each other"* (C1.R1).

The MKX application was thus an asset within the HIM field and given the CEO's high degree of command on that asset, he acquired a high amount of economic capital in the HIM field. Through the implementation of this tool, the CEO was also perceived as an innovation leader. He therefore defended this dominant position within the HIM field and increased his degree of command on the MKX application (economic capital) by changing the user stories at any time during the sprint to ensure that the MKX application contained any necessary functionality which he perceived as important to the group.

The changes in the user stories' contents and priorities were encountered under similar conditions at C2. In C2, these changes were initiated by the customers, who were in dominant positions in the *Durban Team* field, and were perceived as the ones having the final say on all matters pertaining to the Colin application project. The official reason put forward by the customers for such impromptu changes in the user stories was that they only realized mid-sprint that the requirements (put forward by the casino operators) were not correctly understood. However, the empirical analysis revealed that this form of impromptu request also enabled the customers to acquire more economic capital by reasserting their control of the Colin application project. This form of capital was relevant to both the *C2 GASD Team* and the *Durban Team* fields. Consequently, the more capital they held, the more they maintained their dominant positions in both fields.

The empirical analysis showed that the conflict between the team members, as a result of the impromptu changes in the user stories' content and priorities, was accentuated when the software developers were forced to engage in practices that went against what their habitus predisposed them to, while dealing with the changes. In C1, the habitus of the technical experts in the *GASD Software Developers* field predisposed them to expect that the plan which was agreed upon during the sprint planning meeting had to be adhered to during the sprint. They ruled out the possibility of the occurrence of mid-sprint changes to the user stories which had been finalized during the sprint planning meeting and thus never made provisions for such changes while planning. In addition, they resisted when faced with the need to deal with changes to the user stories. When forced to embrace work practices which their habitus did not predispose them to in order to handle the changes mid-sprint, conflict occurred between the team members.

In contrast to C1 where the software developers resented changes to the user stories given their habitus, the software developers at C2 perceived that the changes in the user stories mid-sprint was an acceptable work practice in the team, as can be seen from this extract:

> "*Yes the stories change. Ideally as per sprint principles, you should not change your stories. The scope should not change. But here, we do accept the changes. So that is one deviation that we do from the theoretical Scrum* (C2.R4).

Given their habitus predisposing them to meet the demands of the customers at all costs, the software developers in C2 were predisposed to consider this form of work practice as normal. The software developers also shared habitus predisposing them to plan ahead in the *Pune Team* field but their need to meet the demands of the customers was stronger, hence mitigating the degree of conflict arising because of the impromptu changes in the user stories. However, the software developers still strategized to avoid being blamed in cases of missed deadlines because of these impromptu changes to the user stories. They *put forward disclaimers* (C2.R5) to avoid jeopardizing their positions in the *Pune Team* field whereby they always delivered products on time as can be seen from this extract:

> "*When the user story changes we also highlight the fact that this might delay things. Perhaps 3 or 4 hours extra will be taken for doing this. We highlight how we will implement and how much delay will be included and we start working*" (C2.R3).

7.1.2. Collaboration Breakdowns

The analysis identified various forms of Scrum Process Breakdowns, categorised as collaboration breakdowns, in both C1 and C2. These are discussed in the following sub-sections.

7.1.2.1. Product Owner's Low Level of Authority

According to the Scrum methodology, the Product Owner is responsible for managing the product backlog and deciding on the content of the user stories and their respective priorities, given the input of the customer (Schwaber & Beedle, 2002). However, the empirical analysis revealed that at C1, the CEO (C1.R5) was the key driver behind the product backlog's content and prioritization, and not the Product Owner (C1.R2) as can be seen from this extract:

> *"Sometimes like I said, I really do have specific concerns. It could be certain flow issues, that I don't like that I could see really doesn't work, so I would address that in a very specific way as well. And often, there are some very new stories which were never in the backlog before. So I don't regard the backlog as sort of holy in any way, if things need to be changed, it gets changed and reprioritized"* (C1.R5).

The Product Owner's perception of his ability to manage the product backlog was often that he was in control. However, further observations and questioning indicated that this was not entirely the case, as can be seen from this extract:

> *"Maybe it's not right to say that I've got the final say. He tells me what he really wants to go into the next sprint and, depending on the time we have and my discussion with the team, some requirements might not make it into the sprint"* (C1.R2).

In reality, the Product Owner would always have a private meeting with the CEO prior to any sprint planning meeting. During that meeting, the CEO would inform the Product Owner of the user stories which he would like to have completed in the sprint and their corresponding priorities, as shown from this statement:

> *"C1.R2 is actually very much in contact with C1.R5, who actually makes the call when it comes to user stories content and priorities. C1.R5 will communicate almost on a daily basis with C1.R2, to make sure that C1.R2 knows exactly what is required and what's in the pipeline"* (C1.R3).

In C2, no such form of sprint interruption was encountered. The Scrum Master and the Product Owner, who both held the customer position in *C2 GASD Team* and the *Pune Team* fields, collaborated to identify the user stories' content and priorities. The Scrum Master was also involved in this process given his high amount of economic capital and symbolic capital in both fields. But given the fact that they both held the "customer" position in these fields, it can be said that no breakdowns were experienced, as the Scrum Process was followed as per the methodology's principles.

- ***Conditions leading to the Product Owner's Low Level of authority***

The empirical analysis revealed that the Product Owner (C1.R2) had a low level of authority while deciding on the user stories content and priorities, given the fact that he had limited capital within the C1 GASD field in comparison to the CEO (C1.R5) who held a dominant position and possessed a high amount of symbolic, cultural, and economic capital. C1.R5 had a high amount of economic capital pertaining to his degree of command on the MKX application, a high amount of cultural capital pertaining to his previous role in the C1 GASD team, and a high amount of symbolic capital pertaining to his job title. C1.R2 was new to the team and possessed no economic capital and little cultural and symbolic capital.

C1.R5 claimed that he decided on the MKX application's requirements and priorities based on his perception of needs and the requests that he obtained from the end-users, as can be seen in this extract:

> *"If I perceive something to be particularly wrong in the application, (particularly in terms of the flow in the application), I would prioritise it. If I get a very strong requirement from one customer, for example, the head of MIH internet or some members of the board of executives, I would certainly prioritise that* (C1.R5).

However, requirements were most of the time imposed onto the end-users, as opposed to being obtained from them. This was justified by the CEO as a means to maintain order and the strategic vision of the MKX application, as can be seen from this statement:

> *"I receive some sets of suggestions and feature requests, but one has to be pragmatic about it [...] There's no way where I could practically facilitate a system where anybody could actually put down their requirements. We could put up*

something, saying here's a website go put it down there, it would be chaos. Even though we want this application to be as useful as possible, it's not a free-for-all. So in that sense, we take strategic responsibility of where it's going and drive it in that way" (C1.R5).

The analysis revealed that this form of work practice, which appears to be drawn by a strategic intent of maintaining the MKX application vision was however geared by the CEO's need to retain control over the MKX application. In doing so, he maintained the high degree of economic capital which he possessed in the C1 GASD Team and the HIM fields and reaffirmed his dominant position in both fields.

However, the Product Owner (C1.R2) perceived that it was normal for the CEO to dictate the user requirements and priorities. At times, he even believed that the decisions being made were mutual even though in practice this was not the case. He attributed the CEO's right to dictate the user stories' description and priorities to the latter's degree of experience in the MKX application.

7.1.2.2. Different Perceptions about Task Urgency between the Software Development Sites

The empirical analysis revealed that in C1, the team members had different perceptions about task urgency at the various software development sites. This was particularly felt when user-stories changed unexpectedly mid-sprint, and the changes in the codes had to be implemented urgently to reflect the new requirements and not miss the sprint deadline. This situation can be interpreted as a breakdown when the unwillingness of a team member to treat a change request as urgent results in the sprint goal not being met and the number of tasks scheduled for the sprint not to be completed. In C1, it was noted that even though the Brazilian team members were aware of the new content of the user-stories, they did not acknowledge the sense of urgency in implementing the new user-stories, in contrast to the team members at the South African site, as can be seen from the following statement:

"Obviously the urgency in the South African branch will be properly passed on from person to person because C1.R5 will come in and it's obvious that he wants something done. But the same urgency won't get transferred to Brazil necessarily. That sort of thing does create some problems" (C1.R8).

In C1, the South African team members were physically located in the same office as the MKX application requirements coordinator, enabling them to be at the forefront whenever changes in the user stories' content and priorities were put forward by the latter. Consequently, in spite of the disruption induced by the change in user stories' priority and content, the South African team members nevertheless felt a sense of urgency in completing the new requirements for the sprint, and somehow appeared to be less negatively impacted by the disruptions. However, they felt that they could not completely communicate this sense of urgency to their Brazilian counterparts. The Brazilian team members appeared to be more concerned with the wasted development time as opposed to focusing on completing the new set of requirements. Under these circumstances, Scrum process breakdowns were experienced as team members no longer shared the same purpose, and conflicts and frustration arose. In C2, the sense of urgency arising when user stories' content and priorities changed mid-sprint was easily conveyed to the software developers offshore, despite the fact that they were not located in the same location as the drivers of the project onshore.

- ***Conditions leading to different perceptions about task urgency between the software development sites***

The empirical analysis revealed that this particular form of Scrum process breakdown was particularly linked to the global nature of the team and the resulting nested fields emerging because of the distance. Team members in each nested field had different habitus as well as cultural and symbolic capital, which further induced discrepancies in expectations pertaining to the work practices. For instance, team members within the *Cape Town Team* field had habitus predisposing them to value the need to be output oriented, which, when faced with particular demands and constraints in the field, predisposed them to focus on completing any tasks required of them, at all cost. On the other hand, the team members within the *Sao Paulo Team* field valued precision during coding. In practice, this was reflected in their constant effort in writing optimised algorithms. Consequently, when having to discard software codes in which they had dedicated huge effort, they often did not acknowledge the need to nevertheless urgently complete the new set of requirements and were more focused on the issues of wasted code, thus causing conflict in the team. As such, even though team members belonged to a common C1 GASD field, the differences in habitus deriving from their respective sub-fields given the geographical distance, induced discrepancies in practice.

The influence of habitus on the perception of urgency when user stories changed was corroborated in C2, whereby the agents were predisposed to act on the urgency given their habitus. In particular, software developers in the *Pune Team* field internalised habitus which predisposed them to maintain their productivity level. The importance of productivity in that field predisposed them to adapt their work practices to ensure that no external constraints would negatively affect their productivity level. In addition, software developers were in a dominant position in the *Pune Team* field and were known to deliver high quality products on time. By being in tune with any sense of urgency arising in the team, they were able to take the necessary measures to defend that position, thus maintaining the structure of the field.

7.1.2.3. Disagreements on Suitability of Software Engineering Practices

The empirical analysis showed that during the sprint planning meetings, the team members disagreed on the suitability of some software engineering practices employed by the team. For instance, the South African and the Brazilian PHP software developers often disagreed on the nature of the tasks which the team should be focusing on during the sprint. While the South African team members were focused on completing the user stories quickly and efficiently, the Brazilian PHP developers were constantly asking for more time to refine and refactor the code base in order to create highly optimized and reusable algorithms, as can be seen from this extract:

> *"For instance, yesterday during the spring planning, one programmer requested two days to do a task in order to be able to make it reusable. And they were arguing with the CEO because the latter wanted things to be done quickly because of deadline and he was unwilling to waste one day of a programmer just to write perfect code"* (C1.R2).

This situation gave rise to numerous discussions and debates during the sprint planning meetings. In addition, in spite of reaching a consensus, a feeling of frustration still prevailed among the Brazilian PHP developers based on the strong belief that the team was not engaging in important aspects of software development.

The ceremonies (e.g. planning poker) implemented in Scrum are meant to facilitate or assist the team in reaching a consensus on various user stories and how these user stories are to be completed (e.g. which software engineering practices to focus on). When team members are not able to successfully follow these ceremonies to reach a

common consensus, this could be considered as a breakdown because the particular ceremony was not successful.

No such forms of Scrum process breakdown were identified in C2. In contrast, each user story was broken down into stereotyped tasks and all software developers shared the same opinion on the degree of reusability of the code base as well as the amount of effort and time which they wished to dedicate to that end.

- ***Conditions leading to disagreements on suitability of software engineering practices***

The empirical analysis revealed that this form of Scrum process breakdown occurred when team members belonged to overlapping fields, had internalised different habitus and competed for conflicting capital. In the case of C1, the software developers belonged to three sub-fields with differing habitus and symbolic and cultural capital: *Cape Town Team, Sao Paulo Team* and *Brazilian PHP Software Developers*.

	Habitus		
	Sao Paulo Team	**Cape Town Team**	**Brazilian PHP Software Developers**
A relaxed work atmosphere is important	X		
Precision during coding is important	X		
Communication is important	X		
Cape Town Software Developers are output-oriented		X	
Technical aspect of coding is important			X

Table 7-1 – Comparison of Habitus

As can be seen in Table 7.1, the habitus of the agents in the various nested fields did not overlap and some even conflicted with each other. For example, agents in *Sao Paulo Team* and *Brazilian PHP Software Developers* fields valued the importance of precision during coding and the technical aspect of coding. These habitus, when faced with particular demands and constraints in the field, gave rise to particular forms of work practices which were in line with the interests of these agents (e.g. the constant attempt

to focus on programming tasks relating to code refactoring and optimization, even though these tasks might take longer to be completed). On the other hand, agents from *Cape Town Team* valued the importance of being output oriented and consequently, through that habitus, their work practices were geared towards completing all the user stories as fast as possible. As is the case for all software developers in the main *C1 GASD* field, the software developers in Cape Town also belonged to the *GASD Software Developers* field who valued code quality and standard. This implies that in practice, the Cape Town software developers were more inclined towards favouring the need to complete user stories as fast as possible to maintain their productivity. None of the agents were consciously aware of the forms of habitus which drove their actions. In fact, their habitus was imposed on them and they felt that no other way of behaving was possible other than the standards which they believed in. This gave rise to the conflicts during the sprint planning meetings.

Cultural Capital			
	Sao Paulo Team	**Cape Town Team**	**Brazilian PHP Software Developers**
Knowledge of programming language	X		
Expertise in software development tasks	X		
Degree of expertise on software development tasks for MKX application		X	
Degree of expertise on MKX software application architecture		X	
PHP software development skills			X
Symbolic capital			
	Sao Paulo Team	**Cape Town Team**	**Brazilian PHP Software Developers**
Seniority level	X		
Years of experience in MKX application		X	
Degree of influence within the overall PHP community			X

Table 7-2 – Comparison of Cultural and Symbolic Capital

Table 7.2 shows that the relevant forms of cultural capital within the *Sao Paulo Team* and the *Brazilian PHP Software Developers* fields are the k*nowledge of programming language* and *expertise in software development tasks*, and *PHP software development skills* respectively. The empirical analysis revealed that by trying to push for more attention being paid to code optimization in practice, Brazilian PHP software developers were trying to establish this form of cultural capital as being relevant to the *Software Development* field at C1. In doing so, they also tried to enhance their position of power within the *Software Development* field.

The empirical finding that breakdowns are experienced when software developers belong to overlapping and nested fields with diverging habitus and capital is further corroborated by what was observed in C2. In particular, no such instances of Scrum process breakdown were observed in C2 as the software developers all belonged to the *Pune Team* field. In addition, all the developers were not involved in external programming language forums and never attended conferences on any particular software development practices. This implied they were not subjected to conflicting habitus and competed for the same form of capital.

7.1.2.4. Number of User Stories to be Completed During the Sprint is Imposed on the Team (inability for team to be self-organised)

The empirical analysis showed that the onshore customers imposed the number of user stories which had to be completed in the sprint on the software developers in India. According to the Scrum methodology, the Scrum Master should shield the software developers from such interferences and software developers should be able to voice their opinion when a certain number of tasks are not feasible in the sprint. This is an Instance of a Scrum process breakdown since this Scrum principle was not always adhered to at C2:

> *The one bad thing about our project is that, the decision about the sprint backlog is not made by the offshore team. Ideally, the sprint backlog should be decided by the development team. So the product manager will give us the backlog items and then we decide what features we would take are in this current sprint. But currently these decisions are made by Casper. He does not ask us, out of this ten how many is possible for you over this month or two weeks period, or whether we are able to finish this list or if we want to shortlist. We have to complete everything* (C2.R3).

The software developers at times strategized to explain to the customers that some tasks might take longer than expected and that the required number of stories might not be feasible in one sprint. However, these negotiations often failed as can be seen from this extract:

> *"If it is really not possible to complete everything in one sprint, you have to convince them. But it is not 100% in our hands. Many times, priority is so high and the operators really want these features and we must complete them anyway* (C2.R3).

This type of Scrum process breakdown was not encountered in C1. In C1, the software development team was able to express their views on the feasibility of certain tasks and was listened to, by the Product Owner and the CEO.

- ***Conditions leading to the number of user stories to be completed during the sprint being imposed on the team***

The empirical analysis revealed that this form of Scrum process breakdown appeared when the software developers and the Scrum Master belonged to different fields and had internalised habitus predisposing them to focus on conflicting tasks. For example in C2, the software developers belonged to the *Pune Team* field while the Scrum Master belonged to the *Durban Team* field. In addition, the Scrum Master also held the *Customer* position in the *Durban Team* field. This particular situation led to conflicts of interests and prevented him from fulfilling his Scrum Master role as required by the Scrum methodology, given the conflicting forms of habitus and predispositions arising from the two sub-fields.

In the *Durban Team* field, the Scrum Master was in the position of *customer* and all agents in that position valued the need to retain control over the project and to meet project deadlines. Consequently, he was predisposed to ensure that as many stories as possible were completed in the sprint. Because of such conflicts of interests, he did not shield the team against negative influences from the Product Owner and the other customers who usually requested large numbers of user stories.

The agents in the customer position from the *Durban Team* field also defended their dominant position by imposing the number of user stories to be implemented in the team on the software developers. Even though the customers justified this work practice by the need to complete urgent requests from the casino operators, the empirical analysis

suggested that it was instead a strategy to maintain their dominant position in both the *Durban Team* sub-field and the *C2 GASD Team* field. The software developers could not successfully strategize against this, given their limited amount of relevant symbolic, cultural and economic capital in the *C2 GASD field.* In this case, the economic capital held by the agents in both the C2 GASD Team and Durban Team field were particularly relevant in determining their ability to decide on the number of user stories for one sprint, and the software developers possessed no such capital.

In addition, the software developers were victims of symbolic domination whereby they did not perceive that they could strategize further and acquire more capital, thus enabling them to counter the customers' decision on the number of user stories. Most of the time, they ended up accepting the decisions imposed onto them by the customers and perceived that the situation was necessary in order to meet the customers' demands.

In contrast, in C1 the Scrum Master belonged to the same field as the software developers (i.e. *GASD Software Developers* field) and he shared the same habitus and valued the same forms of capital as the rest of the software development team. In the absence of conflicting forms of habitus, he was able to more easily play his role of Scrum Master and shield the team from influences and inappropriate requests from the Product Owner.

7.1.2.5. Low level of Communication Openness During Meetings Involving the Whole GASD Team

The empirical analysis revealed a form of Scrum process breakdown whereby some team members were less open during meetings involving the whole GASD team, than during internal meetings held at one site. It was observed in C2 that the team members in India express themselves more freely and openly during internal meetings at the offshore site, than during the sprint planning meetings involving the entire team (all meetings were held in English). When meeting with the onshore customers, some software developers did not always freely expressed their opinions unless explicitly probed by agents in more dominant positions (e.g. Project Managers and customers) in the *C2 GASD Team* field. However, these same agents were more open during internal meetings as can be seen from this extract: *During the internal scrum, whenever we have some difficulties, everybody tries to give their opinion and everyone is listened to* (C2.R5). No such forms of Scrum process breakdowns were encountered in C1. Meetings were always held with

team members from both sites, and all agents were seen to openly engage in debates during such meetings.

- ***Conditions leading to a low level of communication openness***

The analysis shows that the degree of openness in a meeting involving the entire team was depended on the amount of relevant capital one had in the joint field, in addition to cultural differences. Team members would be more open in the nested field if they possessed more capital within that field. For example, agents in a dominant position in the *Pune Team* field possessed symbolic and cultural capital which enabled them to easily put their points forward while engaging in debates. These agents were experts in software development tasks for the Colin application and had high academic qualifications (cultural capital). They were also senior team members (symbolic capital). The high amount of capital which they possessed enabled them to debate and be listened to by their team mates during internal meetings. However, their capital was not equally valued in the joint field, and hence they could not discuss openly while engaging in work practices within that field.

During the internal meetings the software developers discussed various issues which included: doubts which they might have about the user requirements, how they could use Scrum statistics (burn-down charts, velocity graphs, bugs reports etc.) to better convince the customers that some tasks were not feasible in the sprint given their manpower capacity, or they discussed how they could better use their particular area of expertise to convince the customer of the required duration of a task. These points were then presented to the customers during the meetings involving the entire team. Hence, even though the lack of open discussions in meetings involving both onshore and offshore team members could be perceived as a Scrum process breakdown, it could also be seen that there was an attempt on the part of the software developers to express themselves. The internal meetings could be seen as a strategy used by the software developers to enhance their positions in the *C2 GASD Team* field by putting forward some of the ideas which they would have identified as a group.

In contrast, the software developers in C1 had enough symbolic and cultural capital to put forward their ideas on the spot, during the meetings with both sites (Cape Town and Sao Paulo). They did not need to resort to internal meetings to consolidate any ideas before expressing them to everyone else.

7.1.2.6. Disagreements on Estimation Mechanisms

Another form of Scrum process was the disagreements between the team members on the estimation mechanisms. During the sprint planning meetings team members at times disagreed on the estimation mechanisms to work out the duration of a task. For example, the duration of tasks for the user stories was estimated in days. However, at the time of the case study, it was decided by the COO (C1.R1) that the number of hours would be used as the estimation mechanism as can be seen from that extract: *"actually C1.R1 as the chief operations officer feels like he needs to have a look at the hours"* (C1.R6).

While the new policy was implemented and being used within the C1 GASD team, the Brazilian and South African team members disagreed on its usability and effectiveness. As mentioned by C1.R6:

> *"Hour estimation is something that does not feel natural to me. I really have problems with hour estimation. Estimating that a task is going to take X hours to do, it feels horrible to me, I can never get it right. We had a long discussion here in Brazil as to whether this was really the proper way to measure the team's process. I personally think that it doesn't really reflect reality since estimating hours is always trial and error. And we usually end up estimating like it's going to take 4, 8, 16, 24. It's not like we say that it's going to take 1.2 hours".*

C1.R1 might be a member of the Scrum team but according to the Scrum methodology, the team should be self-organised and be able to decide on any changes made to the software development process together. This decision cannot be imposed by one team member on the rest of the team.

In C2, task duration was initially estimated in hours and it was then decided that it would be done in days by the customer, as can be seen from that extract:

> *"We estimate in days. Previously it was happening in hours and now it is happening in days. I'm not sure why the change happened. Casper decided on the change* (C2.R4).

However, some of the software developers still felt that they should give their estimates in hours in order to provide more precision:

"I do it in hours in order to get maximum amount of precision. Because if you say one day, then it becomes very difficult to convince the client that you have to spend extra amount of hours on a task. It's easier to say 'I did this for 4 hours' rather than 60% of the day. I feel that giving my estimates in hours is much more convenient" (C2.R6).

The discrepancy and the change in estimation from hours to days did not induce any form of conflict in the team. Most team members accepted the change in estimation mechanism proposed by the customer, and those who did not were not forced to abide by the new standard.

- ***Conditions leading to disagreements on estimation mechanisms***

The empirical analysis revealed that Scrum process breakdowns pertaining to disagreements in estimation mechanisms occurred when team members were forced to engage in practices which conflicted with what their habitus predisposed them to. For example in C1, some software developers from the *C1 GASD Team* field also belonged to the *Sao Paulo Team* field in which they had internalised different habitus. Brazilian software developers from the *Sao Paulo Team* field internalised a habitus calling for the importance of maintaining a relaxed work atmosphere and thus, in practice, having a self-managed team. Estimation in hours appeared to induce a high degree of stress to the Brazilian software developers as it implied the existence of a highly controlled work environment where their work hours were carefully monitored. This contradicted the way they understood the world (their habitus) which valued a relaxed work atmosphere, and thus introduced conflicts in the team.

No such breakdowns were experienced in C2 since all the software developers belonged to the same field - *Pune Team* and all shared the same habitus. In addition, software developers who felt that they had to estimate in hours, in spite of what was requested by the customers, did so in order to defend their position in the *C2 GASD Team* field and the *Pune Team* field. They strategized by providing as accurate estimates as possible, in order to maintain their productivity level and trustworthiness in the eyes of the customer.

7.2. Scrum Process Breakdown Occurring During and After Retrospective Meetings

The Scrum process breakdown instances identified during and after the retrospective meetings have been categorised as *Collaboration Breakdowns*. The two forms of Scrum

process breakdowns under this category, along with the social conditions leading to them, are next described. Appendix 30 summarises the Scrum process breakdowns identified during and after the retrospective meetings and the social conditions leading to them in both cases.

7.2.1. Collaboration Breakdowns

Two forms of Collaboration Breakdowns occurring during and after the Retrospective meetings relate to *decisions on Scrum process updates not made by the development team* and *selective invitations to retrospective meetings*. These are further described below.

7.2.1.1. Decisions on Scrum Process Update not Made by the Development Team

The Scrum methodology requires that during the retrospective meetings the team members reflect on what went well, what went wrong and what should be improved in the sprint. The whole team is responsible for putting forward a list of actionable sets of measures to be followed to improve the Scrum process (Schwaber, 2009). This stems from the agile principle that the agile team should be empowered and at regular intervals, should reflect on how to become more effective, and tune and adjust its behaviour accordingly (Koch, 2005). However, the empirical analysis revealed instances where decisions on how to update the Scrum process were not made by the development team but were rather imposed on them. For example, within the C1 GASD team, decisions on how to improve the Scrum process were taken by the COO (C1.R1). During the retrospective meetings, following a reflection process, both the South African and the Brazilian team members proposed some measures to be implemented for the next sprint. These measures were evaluated by the COO who would then decide on their validity and appropriateness. He would implement some of them or focus on the measures which he had decided himself, as can be seen from this extract:

> *"Well I think that all of us have our own ideas that we come up with. And this gets communicated to the COO and he'll come up with some suggestions. Obviously, because he knows the process the best at this stage, he makes the call about what is changed and how it is changed"* (C1.R4).

In C2 a similar situation was encountered whereby the software development team attempted several times to request for specific Scrum process updates in order to

facilitate their work, but these were not implemented. For example, the request for setting up a test database offshore was refused as can be seen from this statement:

> *"Environment issues in the sense that, at the moment, these people have their code sitting on their machines. But the database is located in the client environment. Then their web services and webservers are still in the client environment. So we have discussed on whether we can have this environment here for testing purposes. But for security reasons, they don't allow"* (C2.R6).

The Scrum process updates which were implemented were mostly geared towards meeting the demands of the customers. For example, the process was once updated to incorporate more code reviews as the customers wanted to ensure the quality of the code and product as can be seen by this statement:

> *"So in that monthly meeting it also happens. Whether documentation is enough or not, whether code review is happening to the required level or not. Last time they decided to incorporate weekly code reviews"* (C2.R8).

- ***Conditions leading to decisions on Scrum process updates not being made by the development team***

The empirical analysis revealed that the development team members were not able to decide on how to update the Scrum process when they lacked capital in comparison to more dominant team members who would then decide on the Scrum process updates. For example, the inability of the C1 GASD team members to impose their decisions on how to update the Scrum Process at C1 was induced by the existence of one agent with a high amount of the relevant forms symbolic capital within the main C1 GASD field and consequently across all the overlapping sub-fields namely C1.R1. In particular, C1.R1 held a high amount of symbolic and cultural capital pertaining to his Scrum implementer role in the team and his Scrum Master certification title, and was thus in a dominant position of power within the *C1 GASD* field. The other team members did not have enough of the relevant symbolic capital to impose their decisions in that matter. However, the team members felt that C1.R1 was entitled to make such decisions, given his high degree of experience in Scrum and the fact that he implemented Scrum for the team. For example, the decision of making use of VersionOne as a Scrum tool was made by C1.R1. The other team members felt that VersionOne was not an ideal tool. They

identified several flaws in the tool, but nevertheless believed that C1.R1 made that decision in the best interest of the team as can be seen from this extract:

> *"I think that there are two products available on the market and C1.R1 who used to be Scrum master evaluated them. He evaluated the tools available at that stage and he chose VersionOne. I'm not sure what the decision behind it was, but it obviously seemed to be the best tool at the time"* (C1.R3).

In C2, a similar explanation held, whereby the customers, being in a dominant position in the C2 GASD Team and the Durban Team fields, sought to retain control over the software development project and process by making decisions on how to improve the Scrum process. The software developers attempted to have some of their ideas implemented, and in doing so sought to enhance their positions in the C2 GASD field. For example they passed on their suggestions to C2.R8, who they hoped would put them forward to the customer, as can be seen from this statement:

> *"We usually do not ask our customer to improve. Suppose one of the guys onsite is not following standards. I don't directly call the customer to tell them that they are not for example upgrading the wiki properly. Because they are our clients and we cannot talk to them like that. I will instead tell to C2.R8. He will also observe and he handles the situation. Life is easier then. So in the improvement meeting he will put that point forward. He will not say that X was saying that. Instead, he will say that in the team, everybody should follow the rules. In that way, it is always listened. Again maturity wise, he is the right person to decide whether this is really a problem or not"* (C2.R3).

However, given their lack of capital, they were not able to ensure that their ideas were implemented or were even considered. C1.R1 thus applied double-meaning strategies towards the other team members while deciding on how to update the Scrum process at C1, in order to maintain his dominant position. In doing so, he also acquired more symbolic capital in the form of being a Scrum implementer. However, this was in a non-deterministic manner and he justified his actions by stating that he was acting in the best interest of the team.

The team members within the C1 GASD team were victims of misrecognition when forms of Scrum process updates were imposed on them. In spite of not fully reaping the benefits of the changes being implemented within the Scrum process, the team members

felt that C1.R1 was entitled to make such decisions, given his high degree of experience in Scrum and the fact that he implemented Scrum for the team.

In C2, a similar explanation held, whereby the customers, being in a dominant position in the C2 GASD Team and the Durban Team fields, sought to maintain their position by applying double-meaning strategies to retain control over the software development project and process. By ensuring that their decisions on how to improve the Scrum process was maintained, they retained their degree of economic capital pertaining to the control of the project and consequently, their dominant positions in both fields.

The customers at C2 were predisposed to strategize as such given their habitus (1) calling for a need to retain control over the software development project and (2) valuing project deadlines and quality. The double-meaning strategies allowed them to further their amount of economic capital pertaining to their degree of control on the Colin application project, as well as their degree of cultural capital as their degree of expertise on the business logic of the Colin application was leveraged.

7.2.1.2. Selective Invitations to Retrospective Meetings

The empirical analysis showed that the retrospective meetings did not always involve the entire team. For example, it was observed that in C2, the software developers were not invited to participate in the retrospective meetings which were held only between management offshore and the customers onshore. This work practice contradicts Scrum work practices which states that all stakeholders with vested interest in the project (particularly the development team) should be allowed to participate in such meetings and should be able to freely express their opinion on what happened in the sprint as can be seen from this statement from (C2.R5): *C2.R8 goes into retrospective meetings with the client. And we are not involved with that* (C2.R5). Instead, C2.R8 gathered information about what went well and what went wrong in the sprint, as well as the software developers' ideas on what should be improved prior to that meeting. He would then decide whether these are valid issues to be passed on to the customers. In contrast, all team members were allowed to participate in the retrospective meetings at C1, and they thus did not experience such a Scrum Process breakdown.

- ***Conditions leading to selective invitations to retrospective meetings***

The reasons provided for not involving the software developers in the retrospective

meetings was that the customers did not want to hurt the software developers' feelings when productivity issues were raised during the course of these meetings: *What they're thinking is, let's say productivity of this person is not good. They think that if they discuss directly with that person, he will feel bad* (C2.R8). However, their behavior could also be attributed to strategies employed by dominant team members to retain their positions and prevent other members from acquiring capital. For example, in C2, customers employed this strategy to maintain their dominant position in the *C2 GASD Team* field and the *Durban Team* field, while management supported this practice in order to maintain their dominant position in the *C2 GASD Team* field.

In particular, given the fact that C2.R8 was able to oversee the whole software development process and be the liaison between the software development team and customers offshore (from a management point of view), he was in a dominant position. He defended that position by acquiring more capital and thus maintained the usefulness of his role in the team. However, he justified this work practice by applying a double-meaning strategy, claiming that he was only following the process that was put forward by the customer pertaining to the retrospective meeting setup. The software developers accepted this retrospective meeting setup as normal: *I don't directly talk to the customers to tell them that they are not for example upgrading the wiki properly. Because they are our clients and we cannot talk to them like that. I will instead tell to C2.R8. He will also observe and he handles the situation [...]. Again maturity-wise, he is the right person to decide whether this is really a problem or not. Last decision is his* (C2.R3).

7.3. Comparison of Scrum Process Breakdowns with Literature

The term "Scrum process breakdown" has not been used in the literature to date. Instead, past studies have spoken about 'social challenges' or 'socio-cultural challenges', both of which point to the existence of Scrum process breakdowns. Consequently, the Scrum process breakdowns identified in this study have been compared to the social challenges identified in the literature. The following sub-sections compare the Scrum process breakdowns identified during and after (1) sprint planning meetings and (2) retrospective meetings. Some of the Scrum process breakdowns identified in the study were similar to what was mentioned in the literature, while others differed, as can be seen in Tables 7.3. A theoretical discussion on the social conditions leading to the breakdowns is provided in Chapter 8.

7.3.1. Findings vs. Literature: Scrum Process Breakdowns Relative to Sprint Planning Meetings

The study showed that team members from the various sites had different perceptions about the degree of urgency of certain tasks. This was particularly visible when tasks were updated because decisions made during the sprint planning meetings about requirements' content and priorities were overruled. This was corroborated by literature, where it was reported that there can be a difference in perception pertaining to the sense of urgency in completing certain tasks in GSD (Clerc, Lago, & van Vliet, 2007). It was also found that there can be a lack of control on the motivation level of the participants at the dispersed sites (Batra, 2009; Batra et al., 2010) and that dispersed team members can have different senses of time (Sutherland et al., 2008).

Study Findings	Literature Findings
Different perceptions about task urgency at the software development sites	Lack of control on motivation at diverse site (Batra, 2009; Batra et al., 2010) Different sense of time (Sutherland et al., 2008) GSD can result in a difference in perceptions pertaining to the sense of urgency in completing some tasks (Clerc, Lago, & van Vliet, 2007)
Disagreements on software engineering practices	Different work styles and need for structure (Sutherland et al., 2007)
Low level of communication openness during meetings involving the whole GASD team compared to internal meetings at the sites	Inability to engage in debates (Summers, 2008) Hard to establish good interactions between the team members (Danait, 2005; Smits & Pshigoda, 2007)
Impromptu changes to user stories' content and priorities	Customers introduce modifications in the middle of a sprint as they are unsure of what they want (Cervone, 2010)
Product Owner's low level of authority	None
Disagreements on estimation mechanisms	None
Number of User Stories to be completed	None

during the Sprint is imposed on the team	
None	Inability to communicate client nuances, context and priorities (Sutherland et al., 2008)
None	Inability to create common vision and strategy (Holmstrom et al., 2006)

Table 7-3 - Findings vs. Literature: Scrum Process breakdowns relative to Sprint Planning Meetings

Another Scrum process breakdown, noted in both the literature and the study, relates to the fact that the team members had divergent perceptions about the suitability of some software engineering practices followed by the team. For instance, the participants of the study disagreed on the nature of the tasks which the team had to focus on during a sprint. The team members also differed in their propensity to be technically-oriented while engaging in the coding process. Literature employed more general terms to describe this phenomenon. For example, Sutherland et al. (2007) reported that given the cultural distance, dispersed team members at times had divergent working styles and need for structure. The divergent propensity to be technical relates to the conflicting working styles of the dispersed team members. It can therefore be said that the findings of this study support existing literature.

Another Scrum process breakdown identified in both the study and the literature is the low level of communication openness of some members during joint meetings. Some team members were less open during meetings involving all the dispersed team members than during internal meetings. Literature has described this phenomenon as the team members limited ability to engage in debates during the sprint planning meetings (Summers, 2008), and the difficulty in establishing good rapport between the dispersed team members (Danait, 2005; Smits & Pshigoda, 2007).

Both the study and literature identified the impromptu changes to the user stories' content and priorities as one of the key challenges of a Scrum team. This occurred when user stories content and priorities, which were agreed upon during the sprint planning meetings, changed in the middle of the sprint. In line with the findings of this study, Cervone (2010), found that customers introduced changes to the user stories in the middle of the sprint, but he attributed this behaviour to the fact that the customers were unsure of what they wanted.

The study also identified forms of Scrum process breakdown which were not reported in the literature review for this study. For example, no studies from the literature review identified social challenges relating to the Product Owner's low level of authority while deciding on user stories content and priorities, during distributed Scrum. The study also identified instances where the number of user stories to be completed by the team was imposed on them and where the team members disagreed on the estimation mechanisms to be employed. None of these were reported in the list of literature sources reviewed for this study.

However, some social challenges which could give rise to Scrum process breakdowns identified in the literature were not identified in the study. In particular, the literature reported that, during GASD, it could be difficult for customers to communicate nuances, context and priorities to the dispersed team members (Sutherland et al., 2008). It was also noted that some GASD project teams find it hard to create a common vision and strategy for the project (Holmström et al., 2006). These forms of breakdown were not encountered in either case. This could be because the CEO (for C1) and the customers (for C2) fully engaged in the Scrum process and project requirement definition phases. What was required of the team was made clear to the dispersed members and further, the project requirements were imposed onto them. In addition, the team members at C2 felt the need to satisfy the customer's demands and requirements at all costs and were thus attentive to the vision and strategy of their customers and the requirements priorities presented to them.

7.3.2. Findings vs. Literature: Scrum Process Breakdowns Relative to Retrospective Meetings

The study identified two forms of Scrum process breakdowns relative to retrospective meetings; both were similar to the social challenges identified in literature. The study found that decisions on how to update the Scrum process were not always made by the team members. Instead, these decisions were made by more dominant stakeholders (e.g. COO in C1 and the customers in C2). While literature did not explicitly report on a similar challenge, it mentioned the existence of power imbalances within GASD teams (Therrien, 2008) and a lack of understanding of the term "self-organisation", derived from one of the agile principle (Batra et al., 2010). So, to some extent, this study offers further empirical examples of this form of Scrum process breakdown.

The second form of Scrum process breakdown identified was the selective invitation to retrospective meetings noted in C2; i.e. not all the team members were invited to the retrospective meetings. This Scrum process breakdown was justified by the claim that the customer did not want to offend anyone when delicate issues were raised during these meetings. Literature reported on similar situations whereby there was sometimes a fear of conflict amongst GASD team members (Uy & Ioannou, 2008) as well as a formal relationship between the vendor and the customer (Batra et al., 2010). It was also reported that it was hard to establish a feeling of trust and good team-spirit amongst the team members and that the dispersed team members did not understand the importance of teamwork (Holmstrom et al., 2006; Uy & Ioannou, 2008; Dorairaj et al., 2011). While none of the social challenges reported by literature specifically related to retrospective meetings within the GASD context, they were similar to the ones experienced during the retrospective meetings observed during the study.

However, one form of social challenge reported by literature was not identified in the study; the lack of control at remote sites (Ramesh et al, 2006; Abdrzeevski, 2007). In contrast, the study found that the customers retained as much control as possible on the project and the Scrum processes followed at the remote sites, especially for C2. In C1, even though the software developers in Brazil were not as tightly managed, the CEO and the COO regularly visited the remote site to ensure that things were running smoothly. Consequently, this study did not confirm a lack of control at remote sites.

Study Findings	Literature Findings
Decisions on Scrum Process Update not made by the Development Team	Agile Principle Issue - Lack of understanding of term self-organisation (Batra et al., 2010) Agile Principle Issue - Power Imbalance (Therrien, 2008)
Selective invitation to retrospective meetings	Agile Principle Issue - Formal relationship between vendor and customer (Batra et al., 2010) GASD Issue - Fear of conflict (Uy & Ioannou, 2008) GASD Issue - Hard to establish a feeling of trust and good team-spirit, and perceive importance of teamwork (Holmstrom et al., 2006; Uy & Ioannou, 2008; Dorairaj et al., 2011)
None	Agile Principle Issue - Lack of control at remote site (Ramesh et al, 2006; Abdrzeevski, 2007)

Table 7-4 - Scrum Process Breakdowns during and as a result of overruled Retrospective meetings

7.4. Chapter Summary

This chapter described the various forms of Scrum process breakdown occurring during and after the sprint planning and retrospective meetings at C1 and C2. The social conditions leading to these breakdowns in C1 and C2 were also presented, drawing on information provided in Chapters 5 and 6, where the positions, habitus and work practices of the various fields inherent to the cases were described. The empirical analysis has revealed six forms of Scrum process breakdowns occurring during and after the sprint planning meetings namely:

(1) Product Owner's low level of authority
(2) Different perceptions about task urgency between the software development sites
(3) Disagreements on suitability of software engineering practices
(4) Number of user stories to be completed during the sprint being imposed on the software developers
(5) Low level of communication openness
(6) Disagreements on estimation mechanisms.

The forms of Scrum process breakdown occurring during and after the retrospective meetings were:

(1) Decisions on Scrum process update not made by the development team
(2) Retrospective meetings not involving the entire team

Following from the empirical observations, these social conditions leading to the Scrum process breakdowns can be regrouped under five key categories namely:

(1) The desire to acquire more capital
(2) Divergent orientations towards capital acquisition
(3) GASD project stakeholders' low level of capital in the joint field
(4) Different beliefs and values because of multiple fields
(5) Agents oriented to value the importance of planning

8. THE FINDINGS: A THEORETICAL DISCUSSION

The empirical analysis has revealed various social conditions which may lead to Scrum process breakdowns relative to sprint planning and retrospective meetings. Following from the empirical observations, these social conditions have been regrouped under two categories, namely:

(1) GASD project stakeholders low level of capital in the joint field

(2) Different beliefs and values because of multiple fields

This chapter discusses these social conditions. In particular, two theoretical propositions have been formulated based on the empirical findings. The propositions have been labelled TP1 and TP2 respectively. Propositions also have sub-propositions (e.g. TP1a and TP1b are sub-propositions of TP1). Evidence from literature has been sought to assess the validity of these theoretical propositions.

This chapter is divided into four sections. Section 8.1 summarises the social conditions giving rise to the Scrum process breakdowns. Each social condition identified from the study, and the theoretical propositions derived for them, are then discussed in Sections 8.2 and 8.3. The chapter is then concluded in Section 8.4

8.1. Summary of Social Conditions Giving Rise to Scrum Process Breakdowns

Table 8.1 summarises the two social conditions and the Scrum process breakdowns which they give rise to. Appendix 31 lists the Scrum process breakdowns and their corresponding social conditions.

Social Conditions	Scrum Process Breakdowns
GASD project stakeholders' low level of capital in the joint field	Number of User Stories to be completed during the Sprint is imposed on the team (TP1a) Product Owner's low level of authority (TP1b) Low level of communication openness during meetings involving the entire GASD team compared to internal meetings at the sites (TP1c) Decisions on Scrum process update not made by the development team during retrospective meetings (TP1d)
Different beliefs and values because of multiple fields	Disagreements on software engineering practices (TP2a) Different perceptions about task urgency at the software development sites (TP2b) Number of User Stories to be completed during the Sprint is imposed on the team (TP2c)

Table 8-1 – Summary of Social Conditions Leading to Scrum Process Breakdowns

8.2. GASD Project Stakeholders' Low Level of Capital in the Joint Field

It has been established in this study that GASD is practiced in a context characterised by overlapping and nested fields of practice. Bourdieu and Wacquant (1992) described these fields of practice as comprising constellations of agents sharing practices and competing for unique forms of capital. It has also been established, in line with Levina and Vaast (2005) that when team members engage in a GASD project, a new joint field is created for which a unique set of economic, cultural and symbolic capital emerges. In addition, the team members' position in the joint field depends on how their capital in the nested and overlapping fields relate to the capital in the joint field (Bourdieu, 1996).

The empirical findings of this study showed that when the GASD team members had a low level of capital in the joint field, various Scrum process breakdowns occurred. For example, those with a low level of capital were not always able to negotiate the number, content and priorities of the requirements. They were also less open during meetings involving the entire team, and few of their ideas on how to update the Scrum process

were implemented. Given these empirical observations, the following theoretical proposition was formulated.

TP1 *When the Software Development Team Members (SDTM) have low level of capital, they will have limited ability to negotiate the requirements number, content, and priorities AND will be less open while communicating AND few of their ideas on how to update the process will be implemented*

The challenges to collaborate in a joint field because of a low level of capital is corroborated by Espinosa, Cummings and Wilson (2003) and Hinds and Bailey (2003) who have also acknowledged that it is difficult to simultaneously bridge multiple boundaries to achieve effective collaboration in global offshore projects. The study has attributed the difficulty in achieving effective collaboration to team members' low level of negotiated capital in the joint field in comparison to their degree of capital in the nested and overlapping fields. When dominant team members in the nested and overlapping fields are unable to establish the relevancy of their capital and position themselves as dominant in the new joint field conflicts arise in the form of Scrum process breakdowns.

This theoretical proposition is also in line with Metiu (2006)'s findings that the existence of nested and overlapping fields and boundaries gives rise to differences between the team members (Metiu, 2006). Meitu (2006) further posits that these differences act as status markers and inhibit collaboration. Other studies have also found that boundaries between overlapping and nested fields are a source of distinction creating status inequalities. For example, Levina and Vaast (2008) found that status differences are a result of the accumulation of capital within and across the fields and hinder collaboration. Levina (2005) further explained that in the context of Information Systems Development, the status differences between the agents are exacerbated by the pre-existing differences in their backgrounds, given the composition of the team and the context of work. In this current study, the status differences relate to the differences in the degree of relevant capital held by team members in the joint field and the nested and overlapping fields, which further lead to their different positions of power. Moreover in this study, when team members attempted and failed to use capital relevant to their nested and overlapping fields in the joint field to have their views accepted, disagreements and Scrum process breakdowns occurred.

The Scrum process breakdowns induced by this social condition are summarized below. The theoretical sub-propositions have also been formulated and corroborated with existing literature.

8.2.1. Number of User Stories to be Completed during a Sprint is Imposed on the Team

According to Scrum, a team should negotiate and decide on the optimum number of requirements or user stories it believes can be successfully completed in an iteration. The team's decision is based on the current content of the product backlog, the information about the latest product increment obtained from the customer, its capacity (manpower) and performance (Schwaber & Beedle, 2002).

However, it was found in the study that the GASD team were not always able to negotiate these requirements and that consequently, a large number of requirements was imposed onto them at the beginning of an iteration. The GASD team members' lack of capital limited their ability to negotiate against the number of requirements imposed onto them. Consequently, when the team members did not have enough capital in the joint field, the number of requirements imposed on the team members was high. This is in line with previous studies which also recognised the strong imbalance of power between participants involved in offshore projects and belonging to different organisations (Levina & Vaast, 2006; Levina, 2006). Levina (2006) further explained that a project in this context can end up being a "market-like transaction" instead of being collaborative. The "market-like transaction" metaphor was particularly applicable to C2 where the degree of collaboration between the customers and the software development team while deciding on the number of requirements to be completed for one iteration was particularly limited. Collaboration was more effective in C1 where the GASD team members all shared the same organisation affiliation. The empirical observations suggest the following theoretical sub-proposition:

TP1a *When SDTM have low level of symbolic, cultural or economic capital, a high number of requirements will be imposed on the team*

For an agile team to negotiate the number of requirements they would be completing during an iteration, they should have some degree of autonomy over planning decisions (Janz, Colquitt, & Noe, 1997). Past studies have attributed the lack of autonomy and ability to self-organise to cultural disparities between project stakeholders (Batra, 2009). However, in line with this finding, other studies have also found that a team can have

limited ability to make certain types of decisions independently when they are pressurised by more dominant individuals (Drury, Conboy, & Power, 2012). In the case of this study, the dominant individuals were the customers in C2 and the CEO in C1. Bourdieu (1990) attributes dominance to high capital acquisition, and consequently subordination to limited capital acquisition, which further supports the findings of this study.

8.2.2. Product Owner's Low Level of Authority

The analysis showed that the authority of the Product Owner depended on his or her economic capital within the joint field as compared to other agents in the field. The less capital the Product Owner had, the less authority he or she had. In line with Levina and Vaast (2008), the study also found that the Product Owner held less capital as he was a newcomer in the joint field (as he recently joint the team) and was thus unable to collaborate on an equal basis with other members of the field.

It was noted in C1 that the Product Owner, in contrast to the CEO who possessed a high degree of economic and cultural capital, was not always able to decide on the requirements' content and priorities. The findings show that this can be attributed to his lack of control of the MKX application (economic capital), and a limited knowledge of the MKX application domain (cultural capital). The CEO was thus the one making requirement related decisions for the team. The empirical observations suggest the following theoretical sub-proposition:

TP1b *When the Product Owner has low level of cultural and economic capital, he/she will have limited ability to decide on the requirements content and priorities*

Scrum's principles recommend that the Product Owner role should be embraced by one person only, and not a committee of people (Schwaber & Beedle, 2002). This principle is based on the premise that when more than one person is responsible for the management of a product backlog's content and priority, conflicting lists might be produced because of conflicting interests, resulting in frustration and confusion in the team (Schwaber & Beedle, 2002). Instances of confused and frustrated team members were encountered in this study, especially when impromptu changes in requirements were enforced on the team in the middle of an iteration because of misunderstandings between the Product Owner and the CEO.

Beedle, Coplien, Sutherland, Østergaard, Aguiar and Schwaber (2010) state that for a Product Owner to embrace his required role in the team efficiently, he or she should have a certain set of qualities. In particular, a Product Owner should have extensive knowledge of the application domain to make informed decisions about the requirements content and priorities. This is in line with the study where it was noted that the designated Product Owner for the team was not able to make such decisions when he lacked cultural capital pertaining to the knowledge of the application domain. Beedle, et al., (2010) further acknowledged that a Product Owner should have the required authority to prioritise the backlog so that maximum business value is achieved. This study has found that authority in that context is inherent to economic capital pertaining to the degree of control of the application project.

8.2.3. Low Level of Communication Openness During Meetings Involving the Entire GASD Team

The GASD team members' low level of capital also impacted on their ability to express themselves during meetings. In particular, the analysis showed that some team members faced difficulties in putting their point across during meetings involving the project stakeholders from all the software development sites. In contrast, they appeared to be more open and confidently voiced their opinions during internal meetings at their local sites. Rogers (1987) defines communication openness as the "message sending and receiving behaviors of superiors, subordinates, and peers with regard to task, personal, and innovative topics" (Rogers, 1987, p. 54). When participants engage in open communication, they should be able to broach topics relative to complaints, suggestions, good ideas, bad ideas, complaints as well as personal opinions (Rogers, 1987).

This form of Scrum process breakdown occurred when team members possessed more symbolic and cultural capital in their nested and overlapping fields in contrast to the joint field (the GASD team). This breakdown also occurred when the symbolic and cultural capital they possessed in their nested and overlapping fields were not recognised as valid in the joint field. When they possessed enough capital in the nested or overlapping fields, they would easily express themselves within these work contexts, but would fail to do so in the joint field because of a lack of relevant capital. This was seen in C2 where some team members possessed low levels of capital in the joint field in contrast to the nested or overlapping fields where their level of capital was higher. In C1, the team members possessed relatively similar levels of capital in the joint, nested and overlapping fields,

facilitating their degree of openness while communicating in the joint field. The details of these empirical observations suggest the following theoretical proposition:

TP1c *When the SDTM have low level of symbolic or cultural capital, they will be less open during communication*

Past GASD studies have also reported that team members in that context can face difficulties in engaging in debates and open communications. However, these studies have mostly attributed this behaviour to organisational structure and language differences. For instance, they stated that some team members remain silent because of a lack of language proficiency (Paasivaara et al., 2008) or because of the organisational structure of some teams (Summers, 2008). Language was not perceived to be an issue in this study as all meetings (both internal and joint meetings) were conducted in English and the team members reported that they understood each other easily.

The findings from this study offer a different perspective and attribute the difficulties in being open during communication to a lack of capital. It has been demonstrated in past studies that a lack of capital or differences in capital ownership between team members lead to status differences in a team (Levina & Vaast, 2008). Edmondson (2003) has stated that when a team is composed of members with different statuses, training, and norms, open communication is impeded, thus confirming the findings from this study. The team investigated in Edmondson (2003)'s study did not involve software development. However, similarly to a GASD team investigated in this study, the members had disciplinary differences. It was also further stated by Dougherty (1992) that when team members belong to different disciplines, communication challenges can emerge.

8.2.4. Decisions on Scrum Process Updates not Made by Software Development Team

As mentioned in Section 8.2.1, Scrum requires the team to identify and implement ways of improving the Scrum process by focusing on the people, relationships, processes and tools relevant to the team (Schwaber, 2009). Most importantly, the methodology expects the Scrum team members to be empowered to make such decisions (Schwaber, 2009).

The GASD development team members in both C1 and C2 were not able to implement their ideas on how to update the Scrum process. Instead, such decisions were imposed

on them by more dominant agents (COO in C1, and customers in C2). Their failure to impose their suggestions was due to their low level of symbolic and cultural capital in the joint field. The details of these empirical observations suggest the following theoretical proposition:

TP1d *When the SDTM have low level of symbolic or cultural capital, few of their ideas on how to update the software development process will be implemented*

Past studies have found that teams face obstacles when trying to make decisions and are unable to participate fully in decision making processes because of the domination of some individuals (Drury et al., 2012). For example, Project Managers was seen to interfere when the team attempted to embark in decision making processes (Blankenship, Bussa, & Millet, 2011), thus preventing the team from achieving true self-management (Tata & Prasad, 2004). In line with these past studies, the domination of project stakeholders (e.g. management and customers) and their influence on the team's decision making process has been identified in this study. However, the domination of these individuals and the subsequent subordination of the team members have further been attributed to a lack of capital. In particular, knowledge, which Bourdieu (1990) sees a type of cultural capital, was seen to impact the team's decision making ability. In support of this finding, other studies have also recognised that a team as a whole (as opposed to one individual member) is better able to make effective decisions, when it has a high degree of cumulated knowledge (Russo & Shoemaker, 1989; Wheeler & Valacich, 1996). This further corroborates the findings from this study where it was found that a lack of capital hinders decision making.

8.3. Different Beliefs and Values Because of Multiple Fields

As previously mentioned, the study has shown that team members, involved in a GASD project using Scrum, engage in multiple fields of practice. These fields can be overlapping, joint or nested. The study has also found that given the habitus which the team members shared within their nested, joint or overlapping fields, a divide may emerge between them when they are pursuing conflicting interests, giving rise to Scrum process breakdowns. This is corroborated by Levina and Vaast (2006) who found that the different fields to which agents belong can shape their practical competencies and interests and can unite and divide them. In particular, the study has found that the different beliefs and values shared by the participants, given the multiple fields in which they inhabit, lead them to have disagreements on the suitability of software engineering

practices, and divergent reactions to urgency. It was seen when team members shared different beliefs and values because they inhabit multiple fields, a high number of requirements can be imposed on the team. Given these empirical observations, the following theoretical proposition can be formulated:

TP2 *When SDTM share different beliefs and values because they inhabit different fields, there will be frequent disagreements on software engineering practices AND divergent reactions to urgency AND a high number of requirements will be imposed onto them*

The study has found that team members internalised different beliefs and values for the different nested and overlapping fields because such beliefs and values are formed out of history, past experiences and socialisation processes (Jarwitz, 2007). In particular, it was noted in the study that in addition to the joint field, the team members also belonged to larger overlapping fields of practices namely the GSD, Agile and Scrum community fields. Their experiences in these fields were thus actively present and also determined the correctness of forms of practices (Bourdieu, 1990). For example, they would favour certain forms of software development practices because of past experiences in other projects or teams. In the joint field, they would thus seek to engage in work practices which felt like common sense (given their beliefs and values from their nested and overlapping fields), but which, when they conflicted with each other, yielded Scrum Process breakdowns.

The Scrum process breakdowns induced by this social condition are summarized below. The theoretical sub-propositions have also been formulated and corroborated with existing literature.

8.3.1. Disagreements on Suitability of Software Engineering Practices

The empirical findings revealed that the GASD team members often disagreed on two forms of software engineering practices to be employed by the team: the GASD team members were found to disagree on (1) the nature of software development tasks to be undertaken by the team (recall Section 8.3) and (2) the estimation mechanisms to be employed by the team, leading to Scrum process breakdowns during collaboration.

The team members often disagreed on the importance of refining software algorithms to achieve reusability (recall Section 8.3). In addition to being induced by GASD team

members' divergent orientation to acquire capital, this form of breakdown occurred when the team members shared different beliefs and values because they inhabited multiple fields. For example, the software developers in C1 belonged to three separate nested fields with non-overlapping and at times conflicting habitus: *Cape Town Team, Sao Paulo Team* and *Brazilian PHP Software Developers*. In *Sao Paulo Team* and *Brazilian PHP Software Developers* fields, the agents valued the importance of precision during coding. On the other hand, agents from *Cape Town Team* valued the importance of being output-oriented and were geared towards completing all the user stories as fast as possible. When faced with the need to collaborate in the joint field, these agents' actions were found to be driven by their beliefs and values from their respective nested fields. They thus pushed for work practices which at times contradicted each other, leading to breakdowns.

Under similar circumstances, the team members also disagreed on the estimation mechanisms to be employed by the team. For example in both cases, it was found that some team members would prefer estimating in hours, while others would prefer estimating in days. Conflict arose when team members felt frustrated that their preferred estimation mechanism was not implemented. This breakdown occurred when team members in overlapped fields internalised habitus predisposing them to have conflicting expectations from the outcome and purpose of the estimation process. These empirical observations suggest the following theoretical sub-proposition:

TP2a *When SDTM share different beliefs and values because they inhabit different fields, they will have frequent disagreements on the suitability of software engineering practices*

Past studies have also acknowledged the impact of team members' beliefs and values acquired from multiple fields on their ability to collaborate in the GSD context. In particular, these studies have found that when team members do not share common beliefs and values, they face various forms of challenges while collaborating, as was seen from the findings from this research. For example, BjØrn and Ngwenyama (2009) use the term "shared meaning" between a project team's participants to denote these common beliefs and values. Shared meaning helps team members make sense of each other's actions and is built over time and via face-to-face interactions. BjØrn and Ngwenyama (2009) further posit that shared meaning can exist at three levels; lifeworld, organisation, and work practices, which are equivalent to the various fields to which the

participants of this study were seen to belong to. In line with the findings from this study, BjØrn and Ngwenyama (2009) posit that when a project team's participants lack shared meanings, they do not understand each other, which in turn results in breakdowns.

Similarly, other studies acknowledged that in the GSD context, cultural differences prevail and there is a lack of shared context and knowledge between the participants (Simons, 2002). It was also found that these cultural differences can lead to inconsistent work practices across the dispersed sites (Holmström et al., 2006). These further corroborate the findings from this study where the GASD team members disagreed on the nature of the software development practices, given their inherent differences emerging from the various fields to which they belonged.

Passos, Braun, Cruzes and Mendonca (2011), specifically studied the impact of practitioners' beliefs on software project practices. In line with the finding from this study, they found that when participants have conflicting beliefs, they disagreed on what estimation mechanism to adopt. In addition, Passos et al. (2011) claimed that the practitioners' beliefs emerged from their current and past project experience, which also further corroborates the findings from this study.

8.3.2. Different Perceptions about Task Urgency at the Software Development Sites

The study also found that there were different perceptions about task urgency at the various software development sites. While some team members belonging to one site would be sensitive to what they felt was an urgent situation and would take necessary measures to complete the tasks quickly, team members at the other site would be less perceptive, thus leading to breakdowns. This form of Scrum process breakdown occurred when the habitus of team members at a site did not predispose them to take the necessary measures to complete any urgent tasks as soon as possible and instead predisposed them to focus on other issues which they believed to be more important. This gave rise to the following theoretical proposition:

TP2b *When SDTM share different beliefs and values because they inhabit different fields, they will have divergent reactions to urgency*

As previously mentioned, other studies have reported the challenge of conveying a sense of urgency in completing certain requests to all the dispersed sites in GSD (Sutherland et al., 2008; Clerc et al., 2007). It was found in this study that this challenge is also experienced when Scrum is employed in the GASD context. While some studies have attributed the difficulties in communicating urgency to the use of inappropriate communication tools (Clerc et al., 2007) (e.g. through the use of asynchronous tools like emails instead of synchronous ones like the phone), others have found that culture plays an important role in it (Egan, Tremaine, Fjermestad, Milewski, & O'Sullivan, 2006).

These studies, undertaken in the field of GSD, have reported that formal cultures tend to have a greater sense of urgency towards abiding by timetable rules and keeping to deadlines, in contrast to informal cultures. These studies have further posited that given that a GSD team can be composed of team members from both formal and informal cultures, conflicting perspectives and attitudes towards urgency, project deadlines and schedules can emerge in that context (Kayworth & Leidner, 2000). This corroborates the findings of this study whereby it was found that the beliefs and values shared by the team members, which also relates to culture, influence their reactions to urgency. Saunders, Van Slyke, and Vogel (2004) have also stated that it is not possible to separate perceptions about time from culture. In addition, in line with the findings from this study, Hall, Waller, Giambatista, and Zellmer-Bruhn (1999), have found that individuals who are prone to being highly conscious about deadlines and urgency tend to attempt and persuade other team members to adhere to deadlines.

8.3.3. Number of User Stories to be Completed during the Sprint is Imposed on the Team

Section 8.2.1 described the Scrum process breakdown pertaining to the number of requirements being imposed on the team. It was mentioned that this type of breakdown occurs when team members have limited capital in contrast to more dominant team members and are thus unable to negotiate against these requests. The empirical findings have revealed that this form of breakdown can also occur when the Scrum Master belongs to multiple overlapping and nested fields in addition to the joint field.

Beedle, et al., (2010) posited that an essential Scrum pattern (i.e. best-practice) is to select a Scrum Master who can enforce the Scrum rules onto the team, and shield the team from external interferences. The empirical findings suggest that when a Scrum Master has internalised beliefs and values from the overlapping and nested fields, other

than the importance of shielding the team members from external interferences (e.g. requests for excess number of requirements), he/she is unable to fulfil his/her role as required by the Scrum methodology (Schwaber, 2009). In such circumstances (particularly in C2), the Scrum Master was seen to face conflicts of interests. Under such circumstances, the team would receive a large number of requests for requirements at the beginning of the sprint. This was particularly relevant in C2 when the Scrum Master shared a different organisational affiliation to the rest of the team. This empirical observation suggests the following theoretical proposition:

TP2c *When the Scrum Master shares different beliefs and values because he/she inhabits different fields, a high number of requirements will be imposed on the team*

In line with the findings of this study, past studies have also found that the value system of an organisation is expressed in the choices made by its members (Whitworth & Beedle, 2007). In addition, Passos et al. (2001) specifically found that the beliefs of software practitioners significantly influence their practices. It is thus relevant to say that the beliefs and values of a Scrum Master, when the latter belongs to a different organisation to the rest of the software development team, might hinder the latter's ability to abide by the agile rules. This can consequently result in a large number of requirements being imposed on the team.

8.4. Chapter Summary

The purpose of this chapter was to summarise the social conditions leading to Scrum process breakdowns. In light of the empirical findings, five theoretical propositions pertaining to these social conditions were presented and corroborated with literature, in order to assess their validity and relevance.

9. CONCLUSION

This chapter concludes the thesis, seeks to demonstrate the importance of the research, highlights the main findings and evaluates the theoretical contributions. The chapter is divided into eight sections. The first section summarises the study. In particular, the problem statement and rationale, research questions and research design are revisited. A summary of the findings is also provided whereby the Scrum process breakdowns and the social conditions under which they occur are reviewed. The second section details the implications of the findings for the distributed agile software development practice while the third section elaborates on the theoretical contributions of the study. In the fourth section, the contributions of the study are evaluated. The fifth section provides a reflective discussion on the research motivation and questions, theory development, case study design, empirical questions and analysis process. Sections Six and Seven highlight some potential limitations on the study and suggest further research respectively. Section Eight highlights the final conclusion to the study.

9.1. Research Summary

9.1.1. Problem Statement and Rationale

It has been established in past literature that Agile methodologies and particularly Scrum are difficult to realise within the GASD context (Paasivaara et al., 2008). This stems from the fact that collaboration is a difficult endeavour during GASD because of the geographical, temporal and cultural distance separating the software project stakeholders (Espinosa et al., 2003; Hinds & Bailey, 2003). Levina and Vaast (2008) have described this work environment as being composed of multiple and overlapping fields, and mention that collaboration is harder under such conditions.

This study reviewed the literature on the application of Scrum and other agile methodologies in GASD between 2006 and 2011. The review noted that, because of Scrum process breakdowns, social challenges emerge when project stakeholders attempt to collaborate across the geographical locations. The existence of these social challenges revealed the need to further investigate the human-centred aspect of collaboration during GASD.

The literature review has also shown a lack of understanding on the reasons for these social challenges. Past studies have mostly focused on describing the social challenges, and on providing mitigating strategies in the form of best-practices, without detailing the

social conditions under which they emerge. The objectives of the study were thus to investigate the Scrum process breakdowns experienced during and after Sprint planning and retrospective meetings. In addition, the social conditions under which these breakdowns were seen to emerge were investigated in light of Bourdieu's Theory of Practice.

9.1.2. Research Questions

This study investigated the following primary and secondary research questions:

Primary Research Question:

- What forms of Scrum process breakdown GASD teams experience during and after sprint planning and retrospective meetings, and what social conditions may lead to these Scrum process breakdowns?

Secondary Research Questions:

- What forms of Scrum process breakdown may GASD teams experience during and after sprint planning meetings?

- What forms of Scrum process breakdown may GASD teams experience during and after retrospective meetings?

- What social conditions may lead to Scrum process breakdowns during and after sprint planning meetings?

- What social conditions may lead to Scrum process breakdowns during and after retrospective meetings?

9.1.3. Research Design

The study was empirical, qualitative and also followed the positivist research paradigm. A case study research method was employed and two case studies in line with Bonoma (1985)'s "drift" and "design" stages of case study design were undertaken to investigate the phenomena of interest and answer the research questions.

The first case (C1) investigated a distributed agile team executing a software project across South Africa (Cape Town) and Brazil (Sao Paulo). In the second case (C2), the team executed an agile software project across India (Pune) and South Africa (Durban).

The site selection was carefully thought out and the results from the first case informed the second case in order to add more richness to the data being gathered.

In both case studies, data was collected through semi-structured interviews, documentation, field notes and direct observation. This wide range of data collection mechanisms allowed for triangulation. A total of 16 interviews and one focus group were conducted with various project stakeholders (seven at C1, nine at C2, one focus group at C2). Twenty-eight observation sessions of Scrum meetings, sprint planning meetings, sprint review meetings, pre-Scrum meetings, and meetings to discuss the way forward in teams (11 at C1 and 17 at C2) were conducted.

The underlying theoretical framework employed for the study was the Theory of Practice (Bourdieu, 1990). The data analysis was divided into four stages, namely: (1) Analysis to identify Scrum Process Breakdowns at C1, (2) Analysis to understand the work practices and the Scrum Process Breakdowns at C1, (3) Analysis to identify the Scrum Process breakdowns at C2, and (4) Analysis to understand the work practices and Scrum Process breakdowns at C2.

9.1.4. Summary of Findings

The study has identified various forms of Scrum process breakdowns that occurred during and after sprint planning and retrospective meetings. Some of these findings corroborated what had already been identified in the past literature while others were new. In particular, the types of Scrum process breakdowns identified during and after sprint planning meetings pertained to collaboration breakdowns and sprint interruptions and are listed below:

- Different perceptions about task urgency at the software development sites
- Disagreements on the suitability of software engineering practices
- Low level of communication openness during meetings involving the entire GASD team, compared to internal meetings at the sites
- Impromptu changes to user stories' content and priorities
- Product Owner's low level of authority
- Disagreements on estimation mechanisms
- Number of user stories to be completed during the sprint is imposed on the team

The types of Scrum process breakdown identified pertaining to the retrospective meetings related to collaboration breakdowns and relates to:

- Decisions on Scrum process updates not made by the development team

Various conditions were seen to lead to the emergence of the Scrum process breakdowns in the GASD context. The specific conditions leading to the particular forms of Scrum process breakdowns, as identified in the cases, have been summarised in Tables 8.1. The overall list of conditions is as follows:

- GASD project stakeholders' low level of capital in the joint field
- Different beliefs and values because of multiple fields

9.2. Implications of Findings for Distributed Agile Software Development Practice

This study revealed frequent disagreements on the suitability of software engineering practices, and perceptions towards urgency in GASD teams. These are due to the different beliefs and values of the team members who simultaneously belong to multiple fields of practice. In light of the findings, it can be posited that agile software development involves team members who possess various beliefs and values given the various fields of practices to which they simultaneously belong. GASD teams should thus be aware and reflective about the beliefs and values of their respective members in order to create a more cohesive and constructive work environment. Thus, instead of being prone to conflicts, the team could become a space where team members' views are valued and promoted. Team members could become more open about their colleagues' opinions and could listen to them in a non-critical manner, allowing for the identification of best possible work practices which could contribute to achieving project success and leveraging the teams' productivity. In doing so, conflicts and frustrations, resulting from team members pushing for the implementation of conflicting software development practices, could be mitigated.

At an individual level, GASD team members should be aware of the various fields of practice in which they personally engage. In doing so, they might become more reflective about whether their actions are geared towards achieving their individual goals as opposed to fulfilling the team goal. This could further raise awareness in the team of the

need for each member to be aware of and work towards the team goal, in order to achieve higher team cohesion (Whitworth & Biddle, 2007; Moe et al., 2008).

Given the principles behind the agile methodologies, an agile team is expected to be self-organised, trusted, supported, reflective, and be given the space to adjust its behaviour at regular intervals in order to increase productivity (Koch, 2005). However, the findings from this study have shown that, due to power imbalances prevailing in GASD teams, it is difficult for agile team members to freely embrace these values.

Power imbalances result in the agile team facing a high number of impromptu change requests mid-sprint, being denied access to resources and opportunities, and a limited ability to be open during meetings. It also impacts on the team's ability to be reflective and to implement innovative ideas on how to update the software development process.

It is therefore important for GASD software teams to be fully aware of the power plays prevailing in their work contexts. In light of the findings where it was found that power imbalances mostly occur because of lack of capital or fights for capital, GASD team members should be reflective about the forms of capital which they could acquire in order to be empowered, enhance their ability to effectively engage in the Scrum work practices, and fully abide by the agile values.

In Appendix 32, a set of best-practices have been proposed to overcome existing social challenges experienced when applying Agile and Scrum work practices. The proposed best-practices take into consideration the social conditions under which particular social challenges are experienced. These best-practices could be validated in future studies to assess their relevance.

9.3. The Contributions of the Study in Terms of Llewelyn (2003) and Gregor (2006)

The study contributed to research by looking at the topic of Scrum usage during GASD from a different perspective to what has been done in most past studies. As demonstrated in the literature review, while most studies have to date solely focused on identifying social challenges experienced during GASD, and on proposing mitigating strategies and "best-practices" to overcome these challenges, few studies have examined the social conditions leading to these challenges or Scrum process breakdowns from a practice perspective. In this study, a practice perspective has allowed for the

identification of local habits, tacit knowledge and assumptions of various social groups engaging in the distributed Scrum projects for the two cases. This perspective also allowed for an understanding of how the various stakeholders involved in the distributed Scrum projects interpreted their world given the shared practices which both constrained and enabled them. The study has sought to produce a theory following the standards of Llewelyn (2003) and Gregor (2006). An evaluation of the theory being proposed based on the guidelines put forward by these two authors are presented below.

9.3.1. The Theoretical Contributions as per Llewelyn (2003) Guidelines

In this study, a theory formulated at Llewelyn (2003)'s fourth level (context-bound theorising settings) was developed. The proposed theory can be categorised as context-bound as it is set in the context of GASD and is particularly geared towards understanding the social conditions leading to breakdowns experienced while engaging in specific Scrum meetings in that context. In particular, the theory attempts to create meaning by exploring the relationship between particular Scrum process breakdowns and the social conditions under which they occur. This is in line with Llewelyn (2003) who stated that the aim of a Level four theory is to provide an understanding of how wider contexts for activities are socially organised, while simultaneously revealing the social conditions under which practices occur.

As per Llewelyn (2003)'s recommendations, the theory was formulated so that it is partially independent from the empirical data, and described the social conditions under which the Scrum process breakdowns occur at a higher level of abstraction from the empirical world. This allowed for a shift in focus to the social conditions under which the team members pursued their projects (Llewelyn, 2003).

9.3.2. The Theoretical Contributions as per Gregor (2006) Guidelines

This thesis proposed a mid-range theory detailing the social conditions giving rise to Scrum process breakdowns in the GASD context based on what was observed in the case studies. These components of the theory based on Gregor (2006)'s guidelines are presented in Table 9.1.

Theory Component	Component Description
Purpose and Scope	The five theoretical propositions derived from this study detail the social conditions under which Scrum process breakdowns during and after retrospective meetings may occur in the GASD context
Constructs	Capital (economic, symbolic, cultural) Requirements requests Access to resources and opportunities Disagreements on the suitability of software engineering practices Communication openness Process Improvement ideas Beliefs and values Reaction to urgency Planning
Means of Representation	Theoretical Propositions
Statement of Relationship	Scrum Process Breakdowns experienced during and after sprint planning and retrospective meetings occur under these conditions: (1) Acquisition of economic capital (2) Divergent orientations towards capital acquisition (3) Limited Capital in the joint field (4) Different beliefs and values because of multiple fields (5) Oriented to value importance of planning
Theoretical Propositions	A set of theoretical propositions as articulated in Chapter 8

Table 9-1 - Components of the mid-range theory on Scrum Process breakdowns in GASD

This study has bridged an important gap in the literature by providing key insights on the social conditions giving rise to Scrum process breakdowns using a practice perspective. The empirical findings forming the basis of the theoretical propositions put forward in this study have been compared to literature in Chapter 8 to demonstrate the existence of such insights.

The plausibility and credibility of the theoretical propositions can be attributed to the fact that they draw on key empirical claims derived from the analysis of data gathered from multiple sources in two case studies. In addition, the empirical claims were backed up by empirical extract from the data gathered. This further contributes to the rigour with which the study was executed (Gregor, 2006). The consistency of the arguments being made can be attributed to the systematic manner in which the analysis was undertaken, whereby the evidence of co-occurrence of the Scrum process breakdowns and the social conditions under which they emerged, was sought for in the data. Finally, the

transferability of the arguments has been proven by comparing and contrasting the findings to other theoretical insights obtained from similar studies employing the Theory of Practice in the software development context. The theoretical propositions have also been formulated in such a way that they could be falsified in future research.

9.4. Evaluation of the Contributions of the Study Using Whetten (1989)'s model

According to Whetten (1989)'s model, seven questions can be posed to evaluate the contributions of a research study: *What is new? So what? Why so? Well done? Done well? Why now? Who cares?* In an attempt to demonstrate that the thesis is making a valid contribution to field of IS research, a brief discussion around Whetten (1989)'s seven questions is proposed. Answers to these questions can be found in several discussion points addressed in various sub-section in this thesis. In order to avoid repetition, the reader will be pointed to these relevant sub-sections.

9.4.1. What is New? Does the Research Make a Significant Value-Added Contribution to the Current Thinking?

The study makes a significant value-added contribution to the current thinking as it has addressed research questions which were found to be both relevant and persisting (see Sections 1.2 and 2.7), which have been answered through the formulation of theoretical propositions which abide by the guidelines of Llewelyn (2003) and Gregor (2006) (see Section 9.3). As per the guidelines of Llewelyn (2003), a context-bound theory, set in the context of GASD, was developed. Given Gregor (2006)' guidelines, a mid-range, descriptive theory was proposed. The study makes a significant contribution because it has bridged an important gap in the literature by providing key insights on the social conditions giving rise to Scrum process breakdowns using a practice perspective. The theoretical propositions have been summarised in Sections 9.1.4 and 9.3.2.

9.4.2. So What? Will the Theory likely Change the Practice of Information Systems Research?

The theoretical propositions derived to answer the research questions have been found to have general implications for the field of distributed agile software development as discussed in Section 9.2. In particular, the theory can assist GASD teams create a more constructive work environment. It could also encourage team members to be more reflective and focus on achieving the team goals as opposed to individual goal, thus leveraging team cohesion.

9.4.3. Why So? Are the Underlying Logic and Supporting Evidence Compelling?

The Theory of Practice was employed as theoretical framework to investigate the research questions. The Theory of Practice focuses on the relationships between people, social structures and the influences present in their interactions (Bourdieu, 1990). This theory was thus ideal to understand and theorise about what people do in their daily lives as a means of identifying the social conditions leading to the emergence of the Scrum process breakdowns. In addition, Bourdieu's theory of practice enabled identification of the differences between the agents involved in distributed Scrum in the GASD context, which also contributed to shedding light on the social conditions leading to the Scrum process breakdowns. Concepts from Bourdieu's Theory of Practice were also useful in formulating a set of empirical questions which guided the data collection, empirical elaboration and theoretical elaboration processes

The research questions and theoretical framework required a thorough understanding of the usage context of distributed Scrum work practices. The case study research method was thus ideal for that purpose. By spending time in the field to engage in discussions with the various project stakeholders and observe various meetings and interactions between them, invaluable insights were gathered and a deep understanding of the particular cases was obtained. In addition the concepts from the frameworks informed the researcher on what to look and probe for during the interviews and observations sessions. In the research methodology chapter (Chapter Four), the research paradigm, method, sampling, unit of analysis and data collection methods chosen have been justified.

9.4.4. Done Well and Well Done? Does the Thesis Reflect Seasoned Thinking?

It is believed that the thesis reflects seasoned thinking because firstly, the researcher has undertaken a systematic review of literature between 2006 and 2011 to identify a valid gap in the body of knowledge (see Chapter Two). The literature review was focused on the topic of social challenges experienced during GASD as it formed the basis of the whole research.

Secondly, the researcher also reviewed literature on the chosen theoretical framework (see Chapter Three). The concepts from Bourdieu's Theory of Practice were explained. In addition, the framework was particularised and a set of empirical questions (designed to guide the data collection, analysis and interpretation process) were provided. Upon

formulating the theoretical propositions in Chapter Eight, the validity of these propositions was also assessed based on theoretical insights obtained from other studies employing Bourdieu's Theory of Practice in a systems development context. This allowed for a verification of the claims being made in the theoretical propositions.

Thirdly, the research approach and method have been applied in a systematic manner and have been documented to allow for replication or corroboration by other researchers. In the research methodology chapter (Chapter Four), a detailed account of the data analysis process was provided. In essence, the concepts from the frameworks were used to interpret, categorise and label the phenomena. Relevant relationships between the concepts were also identified. It is hoped that the stages described in Chapter Four have been documented enough to allow for replication in situations where the Theory of Practice is to be applied.

9.4.5. Why Now? Is the Topic of Contemporary Interest to Scholars in this Area?

It is believed that the topic is of contemporary interest to scholars in the GASD field because it has addressed a set of questions which were found to be both relevant and persisting. The proposed research questions were derived after having undertaken a thorough review of past literature where gaps in the distributed agile research space were identified. The relevance of the research questions was based on the fact that nowadays, both agile software development and distributed software development are popular approaches to software projects and the number of distributed agile projects being undertaken is rising (VersionOne, 2012), hence the need for further research on the matter. In addition, given the fact that agile methodologies integrate social concerns and behavioural concerns into software development (Whitworth & Biddle, 2007), the need to understand the social dynamics at play while applying Scrum in the GASD context is relevant. The persistence of the research question has been demonstrated through a review of the literature between 2006 and 2012 where it has been demonstrated that few studies have focused on examining the social conditions leading to Scrum process breakdowns using a practice perspective and theorised about it.

9.4.6. Who Cares? What Percentage of Academic Readers are Interested in this Topic?

The theory proposed in this research study could be relevant to academic readers interested in the field of GASD, GSD, agile software development and Scrum. The study could also be of relevance to researchers investigating virtual teams. From a theoretical

point of view, the study could be relevant to academic readers interested in Bourdieu's Theory of Practice, as well as those interested in the topic of capital acquisition and retention during project work.

9.5. Researcher's Reflections

This section is devoted to a reflection on what I learnt while undertaking this PHD study. An initial version of this thesis was submitted for examination in January 2012. This current version has been written following the recommended changes put forward by three examiners. The comments put forward by the examiners were crucial in helping me to reflect on my perspectives on my research topic, identify several flaws in various aspects of my study, and point me in the right direction in order to address these shortcomings to the best of my ability. This reflection section is focused on (1) how the research motivation and questions of this study evolved, (2) my understanding and interpretation of Bourdieu's Theory of Practice as a theoretical framework, (3) the case study design and how it could have been improved, and (4) the findings of this study.

9.5.1. Research Motivation and Questions

The research motivation was carefully re-thought in order to produce a compelling and justifiable research contribution. This required careful reflection on the gaps in the initial research motivation. In the previous version of this thesis, I based my research motivation on the claim that practitioners' research lacked scientific rigour and that the best-practice approach which they employed did not take into account the usage context. This research motivation was not clearly articulated and substantiated.

In this current version of the thesis, a different approach was employed to identify a compelling research motivation. In particular, I wanted to investigate the social challenges present while employing the Scrum methodology within the GASD context, but at this stage, had not identified an appropriate research angle to investigate this issue. I thus decided to undertake a thorough literature review around social challenges while engaging in GASD, to understand what current literature had focused on to date on that issue.

I understood how crucial it was to undertake a focused literature review, for one to identify and formulate a compelling research motivation. It was following this literature review that I identified that few studies had been geared towards investigating the social conditions under which Scrum process breakdowns emerged during GASD. I chose the

term "Scrum process breakdown" to represent various forms of deviations from the ideal Scrum process within the GASD setting which lead to conflict and disagreements between the GASD team members. I also chose to focus on sprint planning and retrospective meetings, as I saw from the literature on Scrum that these meetings required rich forms of interactions and negotiations between the team members and were thus ideal mediums to investigate social conditions leading to Scrum process breakdowns.

I was thus able to provide a focused research motivation, backed up with evidence from the thorough and focused literature review which I undertook. In particular, I realised the importance of having a research motivation which stemmed from the gaps which are clearly perceived from the literature.

An important point was raised by the examiners around the question of what would contribute to making this dissertation more "scientific" than practitioners' research. While I do not claim that my study is "superior" to other practitioners' or other scientific studies on GASD, I do believe that my study has contributed to filling a gap in the current literature on GASD. By theorising and formulating a theory on the social conditions leading to particular forms of Scrum process breakdowns, I have provided a theoretically informed understanding of the Scrum process breakdowns and Scrum work practices in GASD, and this had not yet been achieved in current literature as has been demonstrated in Chapter Two. This study therefore does not to claim that the "best-practices" being proposed by current literature are inadequate or inferior. Instead, the study attempted to provide a more theoretical understanding of the social influences occurring during GASD while engaging in Scrum.

I also understood the importance of producing focused and well-articulated sets of research questions based on the research motivation. It is recognised that the initial sets of research questions were too loosely formulated to articulate a concrete answer and provide a compelling contribution. Consequently, after reflection, it was decided that a specific phenomenon of interest had to be identified and Scrum process breakdowns during and after sprint planning and retrospective meetings were found appropriate. The definition of a Scrum process breakdown was also clearly articulated in the literature review and the methodology chapters. Sprint planning and retrospective meetings were chosen given the rich forms of interaction taking place between the participants during these meetings (as is required by the Scrum methodology).

9.5.2. Theory Development

In the initial version of this thesis, I based my analysis on a highly fragmented model of Bourdieu's Theory of Practice. My understanding of Bourdieu's theory has deepened and matured after I spent time re-reading his work, as well as other researchers' application of his theory in their studies. In doing so, the flaws in my initial interpretations became apparent. It became evident that a simpler model, as proposed by Schultze and Boland (2000), was more representative of the essence of the Theory of Practice, enabling me to identify the power plays inherent to the GASD team members' collaboration. This simpler model allowed me to see how practice and structure were mutually constitutive and how they shaped each other in the data.

My understanding of the Theory of Practice also deepened, the more I immersed myself in the data analysis and interpretation process. Concepts which initially were still abstract to me became more real and concrete as I saw their enactment in the behaviour of the agents engaged in GASD for my two case studies. The data analysis and interpretation phase thus sharpened my understanding of Bourdieu's Theory of Practice, while this deepened understanding further served to produce a more precise interpretation of the data. This circular process between data analysis and interpretation on the one hand, and understanding and internalisation of the theory's concepts on the other, was mutually constructive.

The initial version of this thesis also made use of the Work Systems Framework (WSF) for data analysis purposes. However, after reflection, that framework was not found relevant to answer the proposed research questions. In particular, I found that the WSF would not give the social perspective required in order to understand the behaviour of the various groups engaging in Scrum work practices during GASD. The WSF intent did not fit the research purpose as that framework was more geared towards assisting in designing information systems as opposed to understanding the social behaviour of actors in an information system.

9.5.3. Reflections on the Case Study Design

I am fully aware that the main limitation of this study is the case study design and the limited time spent in the field collecting data. An ideal case study should have incorporated the following design elements:

- Following a GASD project for which the Scrum methodology is employed, from the project initiation phase, until the project completion phase. This would have provided a deeper understanding of how the joint field was created, how the various forms of relevant capital were negotiated, and how the joint practices and habitus were produced.

- Spending time at all of the project sites, to obtain a deeper understanding of the work practices of the team members located in Sao Paulo (C1) and Durban (C2).

- Undertaking a third case study in line with Bonoma (1985)'s "prediction" stage to evaluate the theoretical propositions put forward at the end of the design stage. For this third case study, the chosen organisation would have similar dimensions to the organisations chosen for the two case studies already undertaken

- Undertaking a fourth case study in line with Bonoma (1985)'s "disconfirmation" stage to evaluate the limit of the generalisations proposed at the end of the prediction stage. This would serve to evaluate the generalisations made against a broader set of cases.

I acknowledge that the case selection process followed during this study was not entirely purposeful but rather opportunistic. I chose the two organisations because they were available, convenient and willing to participate in the study. However, the characteristics of the GASD project teams in which the case studies were conducted were also appropriate to answer the research questions in a credible way, and allowed me to gather the right type of data which could be interpreted using Bourdieu's Theory of Practice. The characteristics of the project team and the sampling dimensions have been summarised in Table 4.3 in Chapter Four.

It is also acknowledged that, while the case studies were not initially designed and executed with the explicit purpose of identifying Scrum process breakdowns, this phenomenon systematically occurred during the empirical observations. I could observe that forms of Scrum process breakdowns occurred while I was in the field, but did not explicitly label them as such. Hence, in addition to having identified gaps in the literature review, reflecting on the initial sets of findings from the previous version of this thesis also helped to further affirm that the investigation of Scrum process breakdown phenomenon was a valid research angle. I acknowledge that an ideal case study should

have been specifically designed for the purpose of investigating the Scrum process breakdowns and thus further propose that two additional case studies (as previously mentioned) be undertaken to further validate or negate the findings.

9.5.4. Reflections on the Analysis Process

The analysis was an iterative and reflective process. In particular, it required several iterations and deep reflection to obtain a complete understanding of the overlapping fields and the positions, habitus and practices at play within these fields. I found that by writing about these fields and their various work practices and by re-reading my own interpretation of the data, my understanding of the conflict and strategies employed by the various agents became clearer and more precise. I was able to revise and refine this interpretation as my understanding of the concepts from Bourdieu's Theory of Practice became more grounded.

However, I battled with the description of this iterative and reflective phase of the data analysis. It was difficult for me to convey a sense of this iterative cycle without giving to the reader a feeling that the analysis process was top-down.

9.6. Limitations of the Research

In this section, limitations of the research are noted. This research was undertaken using multiple case studies. Ideally, to obtain a deep understanding of the complexities inherent to the usage context, a longitudinal case study would have been ideal. However, because of financial and practical constraints, a longitudinal case study was not feasible. In C1, the project team was constrained by time and was not available to the researcher for more than three weeks. In C2, the researcher could not spend more than three weeks in the field due to financial constraints. However, being aware of the consequences of not being sufficiently immersed in the field, care was taken to be diligent with the capturing of daily field notes on personal impressions of the surroundings. Further details about the data collection techniques employed have been provided in Chapter Four.

Ideally, to obtain a better understanding of the Brazilian culture and context (for C1) time should have been spent at the Sao Paulo office. However, a trip to Brazil was not possible due to financial constraints. To mitigate this limitation, video Skype interviews were undertaken with the Brazilian developers, during which they were not only questioned about their agile work practices but also about their culture, views and opinions. However this did not fully provide the richness which could have been gathered

through a field visit and its resulting impact on the data analysis. In C2, interviews and field trips to the SA site were not feasible due to confidentiality clauses and restrictions put forward by the organisation's top management. In particular, the organisation only agreed to participate in the study provided that their client was not made aware of the researcher's presence at the Indian site. I am aware that this might have introduced some degree on bias in the data. However, I tried to mitigate the negative impact of this limitation by carefully questioning key informants (especially those who visited the Durban site on past occasions) to obtain their input on important issues concerning the South African team members. These inputs were compared to each other and discrepancies were handled through deeper probing and observations of meetings and documentations. Also, even though I did not interview the SA team members, I could nevertheless evaluate their conversations, tone of voice, and approach to the work during the daily Scrum meetings and sprint planning meetings where they were communicating with the Indian team over the speaker phone. These observation sessions were invaluable for me to understand the nature of the relationship between the team members of both sites.

9.7. Future Research Work

It is hoped that future research will be undertaken to assess the validity of theoretical propositions describing the social conditions under which Scrum process breakdowns can be experienced, during and after Sprint planning and retrospective meetings in the GASD context. In particular, the following future research work is proposed:

- A case study in line with Bonoma (1985)'s "prediction" stage to confirm or negate the theoretical propositions put forward at the end of the design stage. For this third case study, the chosen organisation would have similar dimensions to the organisations chosen for the two case studies already undertaken.

- Case studies in line with Bonoma (1985)'s "disconfirmation" stage to confirm/negate the limits of the generalisations proposed at the end of the prediction stage. This would serve to evaluate the generalisations made against a broader set of cases. In particular, this current study could be replicated on a bigger GASD project involving more sites. In-depth longitudinal case studies would be undertaken at all the distributed sites to validate the generalizability of the findings.

- Still part of Bonoma (1985)'s "disconfirmation" stage, the study could also be replicated in a co-located setting, and the results could be compared to identify whether there exist fundamental social differences within the two contexts.

- The study specifically focused on the Scrum methodology. Further research can be undertaken to identify whether process breakdowns occur because of similar social conditions in other forms of agile methodologies like XP or Kanban.

9.8. In Conclusion

This study sought to describe the social conditions under which Scrum process breakdowns may occur in GASD while engaging in Sprint planning and retrospective meetings. It is hoped that the utility of the findings will be established by its uptake amongst the IS community and GASD professionals. To conclude, I would like to assure the reader that the conclusions and knowledge claims made in this study are the result of a thorough and detailed data analysis and triangulation to ensure consistency and reliability of the findings. Keen (1980) mentioned that good IS research should demonstrate rigour and relevance. I hope that the rigour and relevance of the study has been proven throughout this thesis and that the findings will prove useful to agile practitioners and IS researchers.

9. REFERENCES

Abbattista, F., Calefato, F., Gendarmi, D., & Lanubile, F. (2008). Incorporating Social Software into Distributed Agile Development Environments. *Proceedings of the 23rd IEEE/ACM International Conference on Automated Software Engineering*, (pp. 46-51). L'aquila, Italy.

Abbott, A. (1995) Things of Boundaries, *Social Research*, 62(4), p. 857-882

Abrahamson, P., Salo, O., Ronkainen, J., & Warsta, J. (2002). *Agile Software Development Methods.* VTT Publications.

Acciaioli, G. L. (1981). Knowing what you're doing: A review of Pierre Bourdieu's Outline of a Theory of Practice. *Canberra Anthropology*(4), 23-51.

Agile Alliance. (2001). Retrieved 01 01, 2012, from http://www.agilemanifesto.org/principles.html

Alavi, M., Brooke, G., & Carlson, P. (1990). The ecology of MIS research: A Twenty Year Status Review. *Proceedings of the 10th Conference on Information Systems.* Boston, MA.

Ambler, S. (2002). Lessons in agility from internet-based development. *IEEE Software, 19*(2), 66-73.

Ambler, S. (2005). Quality in an agile world. *SQP, 7*(4), 34-39.

Andrzeevski, S. (2007). Experience Report 'Offshore XP for PDA Development'. *Proceedings of the Conference on AGILE 2007*, (pp. 376-381). Washington D.C.

Aron, R., & Singh, J. V. (2005). Getting Offshoring Right. *Harvard Business Review, 83*(12), 135-143.

Aspray, W., Mayadas, F., & Vardi, M. Y. (2006). *Globalisation and offshoring of software. Report of the ACM Job Migration Task Force.* Retrieved 01 01, 2012, from Association for computing machinery.

Babbie, E. (1992). *The Practice of Social Research.* Belmont: Wadsworth.

Batra, D. (2009). Modifying Agile Practices for Outsourved Software Projects. *Communications of the ACM, 52*(9), 143-148.

Batra, D., Xia, W., Van der Meer, D., & Dutta, K. (2010). Balancing agile and structured development approaches to successfully manage large distributed software projects: A case study from the cruise line industry. *Communications of AIS, 2010*(27), 379-394.

Beck, K., Beedle, M., van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., et al. (2001). *Manifesto for Agile Software Development.* Retrieved January 1, 2012, from Agile Alliance: http://AgileManifesto.org

Beedle, M., Coplien, J. O., Sutherland, J., Østergaard, J. C., Aguiar, A., & Schwaber, K. (2010). *Essential Scrum Patterns.* Retrieved 03 18, 2013, from www.hillside.net/plop/2010/papers/beedle.pdf

Begel, A., & Nagappan, N. (2007). Usage and Perceptions of Agile Software Development in Industrial Context: An Exploratory Study. *First International Symposium on Empirical Software Engineering and Measurement*, (pp. 255-264).

Benbasat, I., Goldstein, D. K., & Mead, M. (1987). The Case Research Strategy in Studies of Information Systems. *MIS Quarterly, 11*, 369-386.

Berczuk, S. (2007). Back to Basics: The Role of Agile Principles in Success with Distributed Scrum Team. *Proceedings of AGILE 2007*, (pp. 382-388). Washington D.C.

Berteig, M. (2007). *AgileAdvice.* Retrieved 03 17, 2013, from http://www.agileadvice.com/2007/07/19/linkstoagileinfo/agile-is-not-communism/

Bitner, M. J., Booms, B. H., & Tetreault, M. S. (1990). The Service Encounter: Diagnosing Favourable and Unfavourable Incidents. *Journal of Marketing, 54*, 71-84.

BjØrn, P., & Ngwenyama, O. (2009). Virtual Team Collaboration: Building shared meaning, resolving Breakdowns, and creating Translucence. *Information Systems Journal, 19*(3), 227-253.

Blankenship, J., Bussa, M., & Millet, S. (2011). Managing Agile Projects with Scrum. In *Agile.Net Development with Scrum, Professional and Applied Computing* (pp. 13-27).

Bonoma, T. V. (1985). Case Research in Marketing: Opportunities, Problems, and a Process. *Journal of Marketing Research, 22*, 1999.

Bourdieu, P. (1973). Three Forms of Theoretical Knowledge. *Social Science Information, 12*, 53-80.

Bourdieu, P. (1977). *Outline of a Theory of Practice.* Cambridge: Cambridge University Press.

Bourdieu, P. (1986). The Forms of Capital. (J. G. Richardson, Ed.) *Handbook for THeory and Research for the Sociology of Education*, pp. 241-258.

Bourdieu, P. (1990). *Structures, Habits, Practices.* Stanford, CA: Stanford University Press.

Bourdieu, P. (1992). *The Logic of Practice.* Cambridge, UK: Polity Press.

Bourdieu, P. (1993). *The Field of Cultural Production.* Cambridge: Polity Press.

Bourdieu, P. (1996). Understanding. *Theory, Culture, and Society, 13*, 13-37.

Bourdieu, P. (1998). *On Television and Journalism.* Pluto, London.

Bourdieu, P. (1999). *The Weight of the World.* Cambridge: Polity Press.

Bourdieu, P. (2000). *Pascalian Meditations.* Cambridge: Polity Press.

Bourdieu, P., & Wacquant, L. J. (1992). *An Invitation to Reflexive Sociology.* Cambridge: Polity Press.

Braun, V., & Clarke, V. (2006). Using Thematic Analysis in Psychology. *Qualitative Research in Psychology, 3*(2), 77-101.

Brereton, P., Kitchenham, B. A., Budgen, D., Turner, M., & Khalil, M. (2007). Lessons from Applying the Systematic Literature Review Process within the Software Engineering Domain. *Journal of Systems and Software, 80*(4), 571-583.

Carlile, P. (2002). A pragmatic view of knowledge and boundaries: Boundary objects in new product development. *Organization Science*, 13 (4).

Cassel, C., & Symon, G. (1997). Qualitative Research in Work Context. (C. Cassel, & G. Symon, Eds.) *Qualitative Methods in Organisational Reserch.*

Cervone, H. F. (2010). Understanding Agile Project Management Methods using Scrum. *OCLC Systems & Services International Digital Library Perspectives, 27*(1), 18-22.

Clerc, V., Lago, P., & van Vliet, H. (2007). Global Software Development: Are Architectural Rules the Answer? *Proceedings of the 2nd IEEE International Conference on Global Software Engineering.*

Cocco, L., Mannaro, K., Concas G, & Marchesi, M. (2011). Simulating Kanban and Scrum vs. Waterfall with System Dynamics. *Proceedings of the 12th International Conference on XP*, (pp. 117-131). Madrid, Spain.

Cockburn, H. (2002). *Agile Software Development.* Boston, MA: Addison-Wesley Professional.

Conboy, K. (2009). Agility from first principles: Reconstructing the Concept of Agility in Information Systems Development. *Information Systems Research, 20*(3), 329-354.

Conboy, K., & Fitzgerald, B. (2004). Toward a conceptual framework of agile methods: a study of agility in different disciplines. *Proceedings of the ACM Workshop on Interdisciplinary Software Engineering Research* , (pp. 37-44). Newport Beach, CA, USA.

Cook, T. D., & Campbell, D. T. (1979). *Quasi-Experimentation: Design and Analysis Issues for Field Settings.* Boston, MA: Houghton Mifflin Company.

Corradi, J., Gherardi, S., & Verzelloni, L. (2008). Ten Good Reasons for Assuming a 'Practice Lens' in Organization Studies. *Proceedings of the Organizational Learning, Knowledge and Capabilities conference*, (pp. 1-37). Copenhagen.

Cottmeyer, M. (2008). The Good and Bad of Agile Offshore Development. *Proceedings of the Conference on AGILE 2008*, (pp. 362-367). Toronto.

Curtis, B., Krasner, H. & Iscoe, N. (1988). A field study of the software design process for large systems. *Communications of the ACM, 31,* 1268-1287.

Damian, D., & Zowghi, D. (2003). RE challenges in multi-site software development organisations. *Requirements Engineering, 8*(3), 149-160.

Danait, A. (2005). Agile Offshore Techniques - a Case Study. *Agile Development Conference* (pp. 214-217). IEEE Computer Society.

Dibbern, J., Goles, T., Hirschheim, R., & Jayatilaka, B. (2004). Information systems outsourcing: A survey and analysis of the literature. *The DATA BASE for Advances in Information Systems, 34*(4), 6-102.

Dorairaj, S., Noble, J., & Malik, P. (2010). Understanding the Importanceo of Trust in Distributed Agile Projects: A Practical Perspective. (A. Sillitti, A. Martin, Wang X, & E. Whitworth, Eds.) *XP 2010, 48*, pp. 172-177.

Dorairaj, S., Noble, J., & Malik, P. (2011). Bridging Cultural Differences: A Grounded Theory Perspective. *Proceedings of the 4th India Software Engineering Conference*, (pp. 3-10).

Dougherty, D. (1992). Interpretive Barriers to Successful Product Innovation in Large Firms. *Organization Science, 3*, 179-203.

Drummond, B., & Unson, J. F. (2008). Yahoo! Distributed Agile: Notes from the World over. *Proceedings of the Conference on Agile*, (pp. 315-321). Toronto, Canada.

Drury, M., Conboy, K., & Power, K. (2012). Obstacles to Decision Making in Agile Software Development Teams. *Journal of Systems and Software, 85*(6), 1239-1254.

Dubé, L., & Paré, G. (2003). Rigor in Information Systems Positivist Case Research: Current Practices, Trends, and Recommendations. *MIS Quarterly, 27*(4), 597-636.

Dullemond, K., van Gamaren, B., & van Solingen, R. (2009). How Technological Support can enable advantages of agile Software Development in a GSE setting. *Proceedings of the 4th IEEE International Conference on Global Software Engineering*, (pp. 143-152). Limerick, Ireland.

Edmondson, A. C. (2003). Speaking up in the Operating Room: How Team Leaders Promote Learning in Interdisciplinary Action Teams. *Journal of Management Stdies, 40*, 1419-1452.

Egan, R. W., Tremaine, M., Fjermestad, M., Milewski, J., & O'Sullivan, P. (2006). Cultural Differences in Temporal Perceptions and its Application to Running efficient Global Software Teams. *Proceedings of the IEEE International Conference on Global Software Engineering.*

Eisenhardt, K. M. (1989). Building Theories from Case Study Research. *Academy of Management Review, 4*(4), 532-550.

Espinosa, J. A., Cummings, J. N., & Wilson, J. M. (2003). Team Boundary Issues across Multiple Global Firms. *Journal of Management Information Systems, 19*(4), 157-190.

Everett, J. (2002). Organisational Research and the Praxeology of Pierre Bourdieu. *Organisational Research Methods, 56*(5), 56-80.

Fontana, A., & Frey, J. (2000). The Interview: From Structured Questions to Negotiated Text. (N. Denzin, & Y. Lincoln, Eds.) *Handbook of Qualitative Research*, pp. 645-672.

Gadamer, H. G. (1976). *Philosophical Hermeneutics.* California: University of Californai University Press.

Glaser, B., & Strauss, A. (1967). *The Discovery of Grounded Theory: Strategies of Qualitative Research.* London, UK: Wiedenfeld and Nicholson.

Gopal, A., Mukhopadhyay, T., & Krishman, M. S. (2002). The role of software processes and communication in offshore development. *Communications of the ACM, 45*(4), 193-200.

Green, R., Mazzuchi, T., & Sarkani, S. (2010). Communication and Quality in Distributed Agile Development: An Empirical Case Study. *World Academy of Science, Engineering & Technology, 61*, 322-328.

Gregor, S. (2006). The Nature of Theory in Information Systems. *MIS Quarterly, 30*(3), 611-642.

Gremler, D. D. (2004). The critical Incident Technique in Service Research. *Journal of Service Research*, 65-89.

Grenfell, M., & James, D. (1998). *Bourdieu and Education.* London, UK: Falmer Press.

Guest, G., MacQueen, K. M., & Namey, E. (2011). *Applied Thematic Analysis.* SAGE Publications.

Hall, A., Waller, M. J., Giambatista, R. C., & Zellmer-Bruhn, M. (1999). The Effects of Individual Time Urgency on Group Polychronicity. *Journal of Managerial Psychology, 13*, 244-256.

Hanks, W. (2005). Pierre Bourdieu and the Practices of Language. *Annual Review of Anthropology, 34*, 67-83.

Hardy, C., Lawrence, T., & Grant, D. (2005). Discourse and Collaboration: The Role of Conversations and Collective Identity. *Academy of Management Review, 30*(1), 58-77.

Hazzan, O., & Dubinsky, Y. (2006). Can Diversity in Global Software Development be Enhanced by Agile Software Development? *Proceedings of the 2006 International Workhop on Global Software Development for the Practitioner*, (pp. 58-61). Shanghai.

Herbsleb, J. D., & Mockus, A. (2003). An Empirical Study on Speed and Communication in Globally Distributed Software Development. *IEEE Transactions on Software Engineering, 29*(9), 481-494.

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

Highsmith, J. A., & Cockburn, A. (2001). Agile Software Development: The Business of Innovation. *Computer, 34*(9), 120-127.

Hinds, P. J., & Bailey, D. E. (2003). Out of Sight, Out of Sync: Understanding Conflict in Distributed Teams. *Organization Science, 14*(6), 615-632.

Holmström, H., Fitzgerald, B., Ågerfalk, P. J., & Ó. Conchúir, E. (2006). Agile Practices reduce Distance in Global Software Development. *Information Systems Management, 23*(3), 7-18.

Hossain, E., Babar, M. A., & Verner, J. (2009). Towards a Framework for Using Agile Approaches in Global Software Development. *Lecture Notes in Business Information Processing, 32*, 126-140.

Hossain, E., Bannerman, P. L., & Jeffery, R. (2011). Towards an Understanding of Tailoring Scrum in Global Software Development: A Multi-Case Study. *Proceedings of the International Conference on Software and Systems Process*, (pp. 110-119). Honolulu, USA.

Hundermark, P. (2009). Do Better Scrum: An unofficial set of tips and insights on how to implement Scrum well, Available at http://www.scrumsense.com/wp-content/uploads/2009/12/DoBetterScrum-v2.pdf, accessed on 28th October 2013

Ibarra, H., & Andrews, S. B. (1993). Power, Social Influence, and Sense Making: Effects of Network Centrality and Proximity on Employee Perceptions. *Administrative Science Quarterly, 38*(2), 277-303.

Jain, N. (2006). Offshore Agile Maintenance. *Proceedings of the Conference on AGILE 2006*, (pp. 327-333).

Janz, B. D., Colquitt, J. A., & Noe, R. (1997). Knowledge Worker Team Effectiveness: The Role of Autonomy, Interdependence, Team Development, and Contextual Support Variables. *Personnel Psychology, 50*, 877-904.

Järvinen, P. H. (2004). *Research Questions guiding Selection of an Appropriate Research Method.* Retrieved January 05, 2012, from http://www.cs.uta.fi/reports/dsarja/D-2004-5.pdf

Jarwitz, J. (2007). New Academics Negotiating Communities of Practice: Learning to Swim with the Big Fish. *Teaching in Higher Education, 12*(2), 185-197.

Jarzabkowski, P. (2003). Strategic Practices: An Activity Theory Perspective on Continuity and Change. *Journal of Management Studies, 40*, 23-55.

Jawitz, J. (2008). Learning to Assess in the Academic Workplace: Case Study in the Natural Sciences. *South African Journal of Higher Education, 22*(5), 1006-1018.

Jenkins, R. (2002). *Pierre Bourdieu.* London: Routledge.

Kaplan, B., & Duchon, D. (1988). Combining Qualitative and Quantitative Methods in Information Systems Research: A Case Study. *MIS Quarterly, 12*(4), 571-586.

Kaplan, B., & Maxwell, J. A. (1994). Qualitative Research Methods for Methods Evaluating Computer Information Systems. (J. G. Anderson, C. E. Aydin, & S. J. Jay, Eds.) *Evaluating Health Care Information Systems, Methods and Applications.*

Kayworth, T., & Leidner, D. (2000). The Global Virtual Manager: A Prescription for Success. *European Management Journal, 18*(2), 183-194.

Keen, P. (1980). MIS Research: Reference Discipline and a Cumulative Tradition. In E. McLean (Ed.), *Proceedings of the First Conference on Information Systems*, (pp. 9-18). Philadelphia.

Kelly, G. A. (1951). *The Psychology of Personal Constructs (2 vols).* New York: Norton.

Kirk, J., & Miller, M. L. (1986). *Reliability and Validity in Qualitative Research.* Beverly Hills, CA: Sage Publications.

Kirscher, M., Jain, P., Corsaro, A., & Levine, D. (2001). Distributed eXtreme Programming. *Proceedings of the 2nd International Conference on Extreme Programming and Flexible Processes in Software Engineering (XP2001).* Sardinia, Italy.

Klein, H. K., & Myers, M. D. (1999). A Set of Principles for Conducting and Evaluating Interpretive Field Studies in Information Systems. *MIS Quarterly, 23*(1), 67-93.

Kling, J. L. (1991). Adapting Survey Methods to Study the Social Consequences of Computerisation. (K. L. Kraemer, Ed.) *The Information Systems Research Challenge: Survey Research Methods.*

Koch, A. (2005). *Agile Software Development.* Boston, Massauchusetts: Artech, House.

Kraut, R. E., Fish, R. S., Root, R. W., & Chalfonte, B. L. (1990). Informal Communication in Organisations: Form. Function and technology. (I. S. Oskamp, & S. Spacapan, Eds.) *Human Reactions to Technology: The Claremont Symposium.*

Kraut, R. E., Lewis, S. H., & Swezey, L. W. (1982). Listener responsiveness and the coordination of conversation. *Journal of Personality and Social Psychology, 43*(4), 718-731.

Krishna, S., Sahay, S., & Walsham, G. (2004). Managing cross-cultural issues in global software outsourcing. *Communications of the ACM, 47*(4), 62-66.

Kuutti, K. (1996). Activity Theory as a Potential Framework for Human-Computer Interaction Research. (B. Nardi, Ed.) *Context and Consciousness: Activity Theory and Human-Computer Interaction*, pp. 17-44.

Kym, H., & Park, W. (1992). The Effect of Cultural Fit/Misfit on the Productivity and Turnover of IS Personnel. *Proceedings of Special Interest Group on Computer Personnel Research Annual Conference.*

Lahire, B. (2011). *The Plural Actor.* Polity Press.

Larman, C. (2004). *Applying UML and patterns: An Introduction to Object Oriented Analysis and Design and Iterative Development* (3rd ed.). USA: Prentice Hall.

Larman, C., & Basili, V. (2003). Iterative and Incremental Development: A Brief History. *Computer, 36*(6), 47-56.

Layman, L., Cornwell, T., & Williams, L. (2006). Personality Types, Learning Styles and an Agile Approach to Software Engineering Education. *Proceedings of the 37th SIGCSE Technical Symposium on Computer Science Education*, (p. 1). Texas, US.

Layman, L., Williams, D., Damian, H., & Bures, H. (2006). Essential Communication Practices for eXtreme Programming in Global Software Development Team. *Information and Software Technology, 48*(9), 781-794.

Levina, N. (2005). Collaborating on Multiparty Information Systems Development Projects: A Collective Reflection-in-Action View. *Information Systems Research, 16*(2), 109-130.

Levina, N. (2006). Collaborating across Boundaries in a Global Economy: Do Organizational Boundaries and Country Contexts Matter? *Proceedings of the International Conference on Information Systems*, (pp. 527-542).

Levina, N., & Vaast, E. (2005). The Emergence of Boundary Spanning Competence in Practice: Implications for Implementation and Use of Information Systems. *MIS Quarterly, 29*(2), 335-363.

Levina, N., & Vaast, E. (2006). Turning a Community into a Market: A Practice Perspective on Information Technology Use in Boundary Spanning. *Journal of Management Information Systems, 22*(4), 13-37.

Levina, N., & Vaast, E. (2008, June). Innovating or Doing as Told? Status Differences and Overlapping Boundaries in Offshore Collaboration. *MIS Quarterly, 32*(2), 307-332.

Lindstrom, L., & Jeffries, R. (2004). Extreme Programming and Agile software Development Methodologies. *Information Systems Management Journal, 21*(3), 41-52.

Llewelyn, S. (2003). What Counts as "Theory" in Qualitative Management and Accounting Research? Introducing Five Levels of Theorizing. *Accounting, Auditing, & Accountability Journal, 16*(4), 662-708.

Lyytinen, K. J., & Ngwenyama, O. (1992). What Does Computer Support for Cooperative Work Mean? A Structural Analysis of Computer Supported Cooperations Work. *Accounting Management and Information Technology, 2*(1), 19-38.

Manicas, P. T., & Secord, P. F. (1983). Implications for Psychology of the New Philosophy of Science. *American Psychologist, 38*(4), 399-413.

Markus, L. (1989). Case Selection in Disconfirmatory Case Studies. (J. Cash, & M. Lawrence, Eds.) *The Information Systems Research Challenge*, pp. 20-26.

Maurer, F., & Martel, S. (2002). On the productivity of agile software practices: An industrial case study. *Proceedings of the International Workshop on Global Software Development.* Orlando, Florida, USA.

Maxwell, J. A., Bashook, P. G., & Sandlow, K. J. (1986). Combining Ethnographic and Experimental Methods. (D. J. Fetterman, & M. A. Pitman, Eds.) *Educational Research: A Case Study in Educational Evaluation: Ethnography in Theory, Practice and Politics.*

Metiu, A. (2006). Owning the Code: Status Closure in Distributed Groups. *Organization Science, 17*(4), 418-435.

Miles, M. B., & Huberman, A. M. (1994). *Qualitative Data Analysis: an Expanded Sourcebook.* California: Thousand Oaks.

Miller, A. (2008). Retrieved January 2, 2012, from Distributed Agile Development at Microsoft Patterns & Practices: http://www.ademiller.com/blogs/tech/about-2/

Mockus, A., & Herbsleb, J. (2001). Challenges of Global Software Development. *Proceedings of the 7th International Symposium on Software Metrics.* Napeville, IL, USA.

Moe, N. B., Dingsøyr, T., & Dybå , T. (2008). Understanding Self-Organising Teams in Agile Software Development. *Proceedings of the Australian Conference on Software Engineering*, pp. 76-85.

Montgomery, B., & Duck, S. (1991). *Studying Interpersonal Interaction.* New York: Guilford.

Myers, M. D. (1997). Qualitative Research in Information Systems. *MIS Quarterly, 21*(2), 241-242.

Myers, M. D., & Avison, D. (2002). An Introduction to Qualitative Research in Information Systems. *Qualitative Research in Information Systems, 4*, 3-12.

Myers, M., & Newman, M. (2007). The Qualitative Interview in IS Research: Examining the Craft. *Information and Organisation, 17*, 2-26.

Naidoo, R. (2004). Fields and Institutional Strategy: Bourdieu on the Relationship between Higher Education, Inequality and Society. *British Journal of Sociology of Education, 25*(4), 457-471.

Nardi, B. A. (1996). Activity Theory and Human-Computer Interaction. (B. A. Nardi, Ed.) *Context and Consciousness: Activity Theory and Human-Computer Interaction*, pp. 69-103.

Nardi, B. A., Whittaker, S., & Bradner, E. (2000). Interaction and outaction: Instant Messaging in Action. *Proceedings of the 2000 ACM Conference on Computer Supported Cooperative Work.* Philadelphia, US.

Ngo-The, A., Hoang, K., Nguyen, T., & Mai, N. (2005). Extreme Programming in Distributed Software Development. *Proceedings of International Workshop on Distributed Software Development.* Paris, France.

Ngwenyama, O. (1998). Groupware, Social action and Organisational Emergence: On the Process Dynamics of Computer Mediated Distributed Work. *Accounting, Management and Information Technology, 8*(4), 127-146.

Ngwenyama, O., & Lee, A. (1997). Communication Richness in Electronic mail: Critical Theory and the Contextuality of Meaning. *MIS Quarterly, 21*, 145-167.

Orlikowski, W. J. (2000). Using Technology and Constituting Structures: a Practice Lens for Studying Technology in Organizations. *Organization Science, 11*(4), 404-428.

Orlikowski, W. J., & Baroudi, J. J. (1991). Studying IT in Organisations: Research Approaches and Assumptions. *Information Systems Research, 2*(1), 1-28.

Paasivaara, M., & Lassenius, C. (2004). Using Iterative and Incremental Process in Global Software Development. *Proceedings of the International Workshop on Global Software Development*, (pp. 42-47). Edinburgh, Scotland.

Paasivaara, M., & Lassenius, C. (2006). Could Global Software Development benefit from Agile Methods? *Proceedings of the International Conference on Global Software Engineering*, (pp. 109-113). Costa de Santinho, Brazil.

Paasivaara, M., Durasiewicz, S., & Lassenius, C. (2008). Using Scrum in a Globally Distributed Project: A Case Study. *Software Process: Improvement and Practice, 13*(6), 527-544.

Passos, C., Braun, A. P., Cruzes, D. S., & Mendonca, M. (2011). Analyzing the Impact of Beliefs in Software Project Practices. *Proceedings of the International Symposium on Empirical Software Engineering and Measurement*, (pp. 444-452).

Patton, M. Q. (2002). *Qualitative Evaluation Methods.* Newbury Park, CA: Sage Publications.

Phalnikar, R., Deshpande, S. D., & Joshi, S. D. (2008). Applying Agile Principles for Distributed Software Development. *Proceedings of International Conference on Advanced Computer Control* (pp. 535-539). IEEE.

Polit, D. F., & Beck, C. T. (2004). *Nursing Research: Appraising Evidence for Nursing Practice.* Philadelphia: Wolters Klower/Lippincott Williams & Wilkins.

Radnitzky, G. (1970). *Contemporary Schools of Metascience.* Goteborg: Scandinavian University Books.

Ramesh, B., Cao, L., & Baskerville, R. (2010). Agile Requirements Engineering Practices and Challenges: An Empirical Study. *Information Systems Journal, 20*(5), 449-480.

Ramesh, B., Cao, L., Mohan, K., & Xu, P. (2006). Can Distributed Software Development be Agile? *Communications of the ACM, 49*(10), 41-46.

Reckwitz, A. (2002). Toward a Theory of Social Practices: A Development in Culturalist Theorizing. *European Journal of Social Theory, 5*(2), 243-263.

Repenning, A., Ioannidou, A., Payton, M., Wenming, Y., & Roschelle, J. (2001). Using components for rapid distributed software development. *IEEE Software, 18*(2), 38-45.

Ricoeur, P. (2004). *The Conflict of Interpretations: Essays in Hermeneutics.* Northwestern University Press.

Robarts, J. (2008). Practical Considerations for Distributed Agile Projects. *Proceedings of the Conference on AGILE 2008*, (pp. 327-332). Toronto.

Rogers, D. P. (1987). The Development of a Measure of Perceived Communication Openness. *The Journal of Business Communication, 24*(4), 53-61.

Russo, J., & Shoemaker, P. (1989). *Decision Traps: Ten Barriers to Brilliant Decision Making and How to Overcome THem.* New York: Simon & Schuster.

Saunders, C., Van Slyke, C., & Vogel, D. R. (2004). My Time or Yours? Managing Time Visions in Global Virtual Teams. *The Academy of Management Executive, 18*(1), 19-31.

Schuh, P. (2005). *Integrating Agile Development in the Real World.* Hingham MA: Charles River Media Inc.

Schultze, U., & Boland, J. (2000). Knowledge Management Technology and the Reproduction of Work Practices. *Journal of Strategic Information Systems, 9*, 193-212.

Schummer, T., & Lukosch, S. (2008). Supporting the Social Practices of Distributed Pair Programming. *Lecture Notes in Computer Science, 5411*.

Schwaber, K. (2009). Scrum Guide. Retrieved 03 18, 2013, from www.itemis.de/binary.ashx/~download/.../scrum-guide.pdf

Schwaber, K., & Beedle, M. (2002). *Agile Software Development with Scrum.* Upper Saddle River, NJ: Prentice-Hall.

Schwandt, T. A. (1997). *Qualitative Inquiry: A Dictionary of Terms.* London: Sage Publications.

Sewchurran, K. (2008). Toward an Approach to Create Self-Organizing and Reflexive Information Systems Project Practitioners. *International Journal of Managing Projects in Business, 1*(3), 316-333.

Sheth, B. (2009). Scrum 911! Using Scrum to Overhaul a Support Organization. *Proceedings of the Conference on AGILE 2009*, (pp. 74-78).

Shrivastava, S. V., & Date, H. (2010). Distributed Agile Software Development: A Review. *Journal of Computer Science and Engineering, 1*(1), 10-17.

Simons, M. (2002). Internationally Agile. *InformIT, 15*(3).

Singer, J., Lethbridge, T., Vinson, N., & Anquetil, N. (2010). An Examination of Software Engineering Work Practices. *Proceedings of the 1997 Conference of the Centre for Advanced Studies on Collaborative Research*, (pp. 21-22). New York, USA.

Sison, R., & Yang, T. (2007). Use of Agile Methods and Practices in the Philippines. *Proceedings of the14th Asia-Pacific Software Engineering Conference*, (pp. 462-469).

Smits, H., & Pshigoda, G. (2007). Implementing Scrum in a Distributed Software Development. *Agile*, (pp. 371-375). Washington DC, US.

Sone, S. P. (2008). *Mapping agile project management practices to project management challenges for software development.* Retrieved January 1, 2012, from www.scrumalliance.org/resource_download/412

Stettina, C. J., & Heijstek, W. (2011). Five Agile Factors: Helping Self-Management to Self-Reflect. *Proceedings of the European Software Process Improvement Conference*, (pp. 84-96). Roskilde Denmark.

Strauss, A., & Corbin, J. (1998). *Basics of Qualitative Research: Techniques and Procedures for Developing Grounded Theory.* Thousand Oaks, CA: Sage Publications.

Stuart, I., McCutcheon, D., Handfield, R., McLachlin, R., & Samson, D. (2002). Effective Case Research in Operations Management: A Process Perspective. *ournal of Operations Management, 20*(5), 419-433.

Summers, M. (2008). Insights into an Agile Adventure with Offshore Partners. *Proceedings of the Conference on Agile*, (pp. 333-338). Toronto, Canada.

Sutherland, J., Schoonheim, G., Kumar, N., Pandey, V., & Vishal, S. (2009). Fully Distributed Scrum: Linear Scalability of Production between San Francisco and India. *Proceedings of AGILE*, pp. 277-282.

Sutherland, J., Schoonheim, G., Rustenburg, E., & Rijk, M. (2008). Fully Distributed Scrum: The Secret Sauce for Hyperproductive Offshored Development Teams. *Agile Conference*, (pp. 339-344). Toronto, Canada.

Sutherland, J., Viktorov, A., Blount, J., & Puntikov, N. (2007). Distributed Scrum: Agile Project Management with Outsourced Development Teams. *Proceedings of 40th Annual Hawaii International Conference on System Sciences*, (p. 274a). Waikoloa, HI.

Swartz, D. (1997). *Culture and Power: The Sociology of Pierre Bourdieu.* Chicago: University of Chicago Press.

Tas, J., & Sunder, S. (2004). Financial services - Business process outsourcing. *Communications of the ACM, 47*(5), 50-52.

Tata, J., & Prasad, S. (2004). Team Self-Management, Organizational Structure, and Judgements of Team Effectiveness. *Journal of Management Issues, 16*(2), 248-265.

Taylor, C. (1976). Hermeneutics and Politics. (P. Connerton, Ed.) *Critical Sociology, Selected Readings*, pp. 153-193.

Taylor-Powell, E., & Steele, S. (1996). *Collecting Evaluation Data: An Overview of Sources and Methods.* Madison, WI: University of Wisconsin Press.

Therrien, E. (2008). Overcoming the Challenges of Building a Distributed Agile Organsiation. *Proceedings of the Conferece on Agile 2008*, (pp. 368-372). Toronto.

Turner, S. (1994). *The Social Theory of Practices: Tradition, Tacit Knowledge and Presuppositions.* Chicago, US: University of Chicago Press.

Urdangarin, R., Fernandes, P., Avritzer, A., & Paulish, D. (2008). Experiences with Agile Practices in the Global Studio Project. *Proceedings of the International Conference on Global Software Engineering*, (pp. 77-86). Bangalore.

Uy, E., & Ioannou, N. (2008). Growing and Sustaining an Offshore Scrum Engagement. *Proceedings of the Conference on AGILE 2008*, (pp. 345-350). Toronto.

Van de Ven, A. H. (1989). Nothing is quite so practical as a good theory. *Academy as Management Review, 14*, 486-489.

Vax, M., & Michaud, S. (2008). Distributed Agile: Growing a Practice Together. *Proceedings of the Conference on AGILE 2008*, (p. 310). Toronto.

VersionOne. (2010). *Fifth State of Agile Development Survey Results.* Retrieved December 19, 2011, from http://www.versionone.com/state_of_agile_development_survey/10/page3.asp

Voss, C., Tsikriktsis, N., & Frohlich, M. (2002). Case Research in Operations Management. *International Journal of Operations & Production Management, 22*(2), 195-219.

Walsham, G. (2002a). Cross-cultural Sofware Production and Use: A Structurational Analysis. *MIS Quarterly, 26*(4), 359-380.

Walsham, G. (2002b). Interpretive Case Studies in IS Research: Nature and Method. (M. D. Myers, & D. Avison, Eds.) *Qualitative Research in Information Systems.*

Walsham, G., & Sahay, S. (1999). GIS for district-level administration in India: Problems and Opportunities. *MIS Quarterly, 23*(1), 39-65.

Webb, E. J., Campbell, D. T., Schwartz, R. D., & Sechrest, L. (1966). *Unobtrusive Measures: Nonreactive measures in the Social Sciences.* Chicago: Rand McNally.

Webb, J., Schirato, T., & Danaher, G. (2002). *Understanding Bourdieu.* London, UK: SAGE.

Weber, R. P. (1985). *Basic Content Analysis.* London: Sage.

Wenger, E. (1998). Communities of Practice. Learning as a Social System. *Systems Thinker.*

Wernick, P., & Hall, T. (2004). Can Thomas Kuhn's Paradigms Help Us Understand Software Engineering? *European Journal of Information Systems, 13*(3), 235-243.

Wheeler, B. C., & Valacich, J. s. (1996). Facilitation, GSS, and Training as Sources of Process Restrictiveness and Guidnace for Structured Group Decision Making: An Empirical Assessment. *Information Systems Research, 7*(4), 429-450.

Whetten D.A. (1989). What constitutes a theoretical contribution? *Academy of Management Review*, 14(4), 490-495

Whitworth, E., & Biddle, R. (2007). The Social Nature of Agile Teams. *Proceedings of the AGILE Conference 2007*, (pp. 36-36). Washington D.C.

Williams, W., & Stout, M. (2008). Colossal, Scattered, and Chaotic (Planning with a Large Distributed Team). *Proceedings of the Conference on AGILE 2008*, (pp. 356-361). Toronto.

Yadav, V., Adya, M., Nath, D., & Sridhar, V. (2007). Investigating an 'Agile-Rigid' Approach in Globally Distributed Requirements Analysis. *Proceedings of the Pacific Asia Conference on Information Systems.*

Yin, R. K. (2003). *Case Study Research, Design and Methods.* Beverly Hills, CA: Sage Publications.

Young, C., & Terashima, H. (2008). How did you Adapt Agile Processes to our Distributed Development. *Proceedings of the AGILE Conference 2008*, (pp. 304-309). Toronto.

APPENDIX 1: List of Papers Reviewed on Distributed Agile Software Development

Paper Code	Paper Reference	Year
[S1]	Hazzan & Dubinsky, 2006)	2006
[S2]	Batra, 2009	2009
[S3]	Hossain, Bannerman, & Jeffery, 2011	2011
[S4]	Holmström, Fitzgerald, Ågerfalk, & Conchúir, 2006	2006
[S5]	Paasivaara, Durasiewicz, & Lassenius, 2008	2008
[S6]	Batra, Xia, VanderMeer, & Dutta, 2010	2010
[S7]	Green, Mazzuchi, & Sarkani, 2010	2010
[S8]	Ramesh, Cao, Mohan, & Xu, 2006	2006
[S9]	Phalnikar, Deshpande, & Joshi, 2008	2008
[S10]	Danait, 2005	2005
[S11]	Dullemond, van Gameren, & van Solingen, 2009	2009
[S12]	Bjørn & Ngwenyama, 2009	2009
[S13]	Sutherland, Schoonheim, Rustenburg, & Rijk, 2008	2008
[S14]	Sutherland, Viktorov, Blount, & Puntikov, 2007	2007
[S15]	Paasivaara & Lassenius, 2006	2006
[S16]	Cannizo, Marcionetti, & Moser, 2008	2008
[S17]	Abbattista, Calefato, Gendarmi, & Lanubile, 2008	2008
[S18]	Whitworth & Biddle, 2007	2007
[S19]	Dorairaj, Noble, & Malik, 2010	2010
[S20]	Dorairaj, Noble, & Malik, 2011	2011
[S21]	Shrivastava & Date, 2010	2010
[S22]	Williams & Stout, 2008	2008
[S23]	Drummond & Unson, 2008	2008
[S24]	Vax & Michaud, 2008	2008
[S25]	Layman, Williams, Damian, & Bures, 2006	2006
[S26]	Smits & Pshigoda, 2007	2007
[S27]	Berczuk, 2007	2007
[S28]	Summers, 2008	2008
[S29]	Cottmeyer, 2008	2008
[S30]	Therrien, 2008	2008
[S31]	Canfora, Cimitile, Di Lucca, & Visaggios, 2006	2006
[S32]	Yadav, Adya, Nath, & Sridhar, 2007	2007
[S33]	Jain, 2006	2006
[S34]	Andrzeevski, 2007	2007
[S35]	Sison & Yang, 2007	2007
[S36]	Young & Terashima, 2008	2008
[S37]	Robarts, 2008	2008

[S38]	Uy & Ioannou, 2008	2008
[S39]	Edwards, 2008	2008
[S40]	Urdangarin, Fernandes, Avritzer, & Paullsh, 2008	2008
[S41]	Sheth, 2009	2009
[S42]	Chubov & Droujkov, 2009	2007
[S43]	Hossain, Babar, & Verner, 2009	2009
[S44]	Schummer & Lukosch, 2008	2008
[S45]	Miller, 2008	2008
[S46]	Berteig, 2007	2007

APPENDIX 2: Summary of Social Challenges Experienced while Applying Agile Principles in the GASD Context

Agile Principles	Distance	Social Challenges	Mitigating Practices
Motivated team members	- Geographical distance	- Lack of control on motivation at diverse site (Batra, 2009; Batra et al., 2010) - No personal relationship to build on (explanation for lack of motivation) (Phalnikar et al., 2008)	- Short iteration - Frequent build - Continuous integration (Dullemond et al., 2009)
Self-organising teams	- Socio-cultural distance	- Lack of understanding of term "self-organisation" (Batra et al., 2010) - Cultural disparities (Batra, 2009) - Formal relationship between vendor and customer (Batra et al., 2010)	- Record meetings (Danait, 2005) - Use of facilitator (Therrien, 2008) - Management support (Berczuk, 2007)
Face-to-face conversations	- Geographical distance - Temporal distance	- Not clearly stated	- Communication and collaboration tools (Danait, 2005; Phalnikar et

				al., 2009; Batra, 2009) - Synchronised work hours - Balanced coordination - Constant communication (Ramesh et al., 2006) - Informal communication (Blog & Documentation) (Ramesh et al., 2006; Abbatisto et al., 2008) - Define customer role (Layman et al., 2006) - Continuous code integration (Shrivastava & Date, 2010)
Working software	- Socio-cultural distance	- Clashing cultural values (Dullemond et al., 2009; Berteig, 2007)	- Not mentioned	
Close collaboration	- Geographical distance	- Lack of control at remote site (Ramesh et al., 2006; Andrzeevski, 2007)	- Customer visits (Andrzeevski, 2007)	
Welcome changes in requirements	- Socio-cultural distance	- Inability to negotiate contract (not explained why) (Batra et al., 2010)	- Not stated	

		- Lack of communication (Paasivaara & Lassenius, 2006)	
Reflection	- Socio-cultural distance - Geographical distance	- Cultural clashes (Batra, 2009) - Power imbalance (Therrien, 2008)	- Empower team (Dullemond et al., 2009)

APPENDIX 3: Summary of Social Challenges Experienced while Applying Scrum work practices in the GASD Context

Scrum Practice	Distance	Social Challenges	Mitigating Practices
Sprint Planning Meeting	- Socio-Cultural Distance - Geographical Distance	- Inability to recognise speakers (Paasivaara et al., 2008) - Differences in Language and culture (Drummond & Unson, 2008) - Inability to engage in debates (Summers, 2008) - Lack of cultural understanding (Drummond & Unson, 2008) - Hard to establish good interaction between team members (not explained why) (Danait, 2005; Smits & Pshigoda, 2007)	- Use of a facilitator (Summers, 2008) - Use of communication tools (Danait, 2005) - Bring people together during early stages of project (Smits & Pshigoda, 2007) - Take holidays into consideration while planning (Hossain et al., 2011)

Daily Stand-up meetings	- Socio Cultural Distance	- Cultural differences and misunderstandings (Paasivaara et al., 2008) - Tense participants (Young & Terashima, 2008)	- Greeting period prior to beginning of meetings (Young & Terashima, 2008) - The use of user-stories (Summers, 2008) - Email answers to three daily stand-up meetings prior to the time (Paasivaara et al., 2008, Sutherland et al., 2007)
Retrospective & Sprint Review meetings	- Socio-cultural distance	- Not effective but no theorisation about why this is the case	- Informal conversations (Shrivastava & Date, 2010)

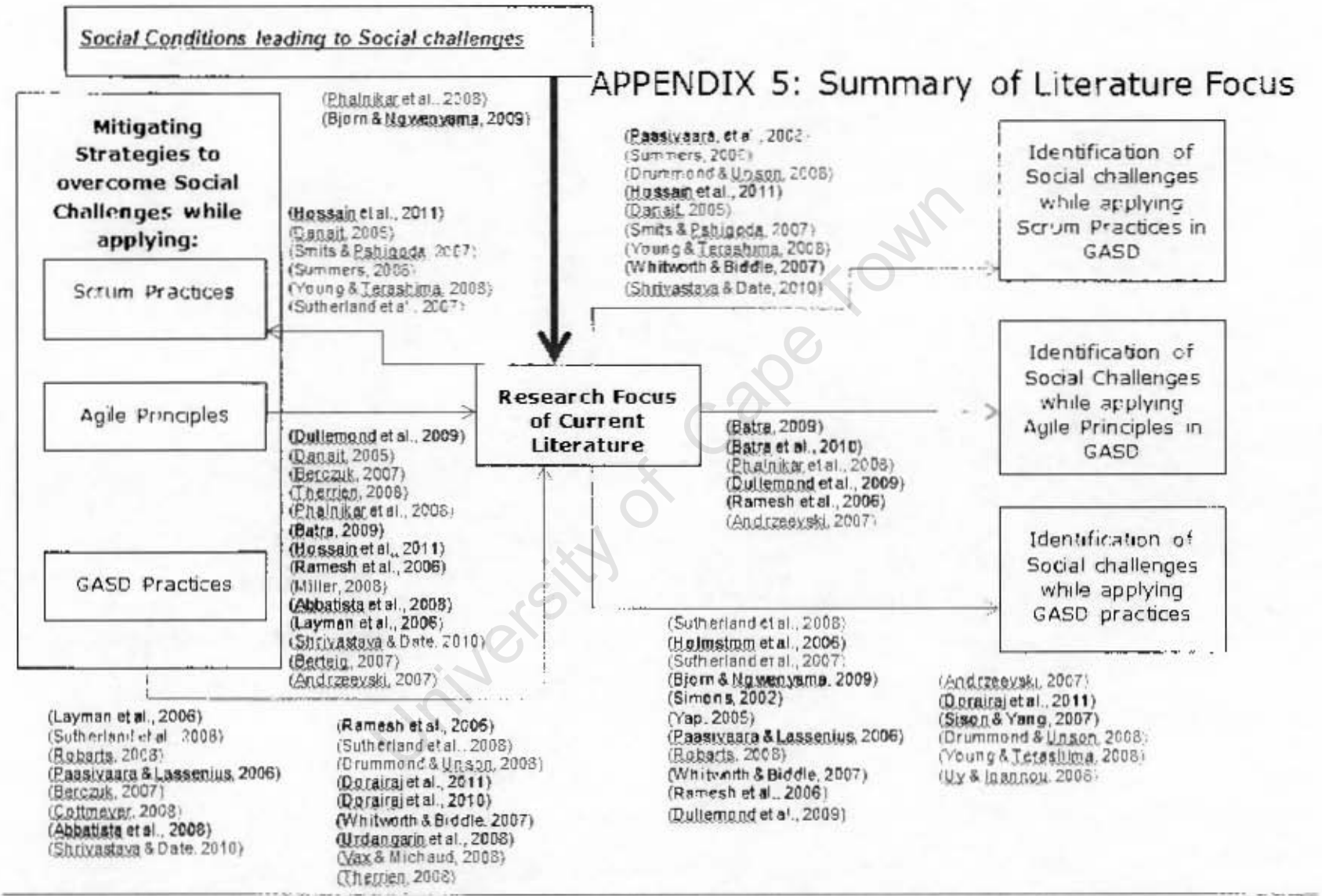
APPENDIX 4: Summary of Social Challenges Experienced while Common GSD Work Practices in the GASD Context

GASD	Distance	Social Influence	Mitigating Practice
Communication Practices	- Socio-cultural distance - Temporal distance - Geographical distance	- Varying attitude towards hierarchy (Holmstrom et al., 2006; Sutherland et al., 2008) Different sense of time (Sutherland et al., 2007) Different communication style (Sutherland et al., 2007) - Different need for structure (Sutherland et al., 2007) - Different work style (Sutherland et al., 2007) - Lack of shared meaning (Bjorn & Ngwenyama, 2009)	- Team member representation at dispersed site (Layman et al., 2006) - Prompt and conclusive response to email (Layman et al., 2006) - One point of contact between team members and customers (Andrzeevski, 2007) - Flexitime (Dorairaj et al., 2011) - Addressing language barrier (Vax & Michaud, 2008; Therrien, 2008) - Increasing formal and informal communication (Dullemond et al., 2009) - Use of technology (Dullemond et al., 2006) - Promote culture of openness (Sutherland et al., 2008)

Coordination practices	-	Geographical distance	- Lack of informal contact (Holmstrom et al., 2006) - Inability to communicate client nuances, context and priorities (Sutherland et al., 2008) - Cultural Differences (Simons, 2002; Yap, 2005) - Inconsistent work practices at diverse sites (Robartz, 2008, Holmstrom et al., 2006) - Inability to create common vision and strategy (Holmstrom et al., 2006) - Various forms of implicit assumption from the team members (Paasivaara & Lassenius, 2006)	- Regular travel (Sutherland et al., 2008) - Always-on Skype connection (Sutherland et al., 2008) - Use Project News Gazette (Sutherland et al., 2008) - Understand motivation (Robarts, 2008) - Use Unit Test Script (Berczuk, 2007) - Divide work into modules (Paasivaara & Lassenius, 2006)
Team cohesion	-	Geographical distance	- Hard to establish a feeling of trust and "teamness" (Holmstrom et al., 2006) - Lack of understanding of	- Regular travel of distributed partners and sponsors (Cottmeyer, 2008; Ramesh et al., 2006; Sutherland et al., 2008; Dorairaj et al., 2010)

		importance of team work (Uy & Ioannou, 2008; Dorairaj et al., 2011) - Fear of conflict (Uy & Ioannou, 2008) - Lack of commitment (Uy & Ioannou, 2008) - Avoidance of accountability (Uy & Ioannou, 2008) - Inattention to results (Uy & Ioannou, 2008) - Different perceptions of hierarchy (Holmstrom et al., 2006)	- Use social network profile (Abbatista et al., 2008) - Establish a safe work environment (Cottemeyer, 2008) - Show respect and support towards each other (Vax & Michaud, 2008; Cottmeyer, 2008)
Knowledge sharing	- Geographical distance	- Not mentioned	- Product repository (Shrivastava & Date, 2010) - Team members rotation (Cottmeyer, 2008) - Pair programming (Urdangarin et al., 2008) - Provide feedback to entire team (Whitworth & Biddle, 2007)

APPENDIX 5: Summary of Literature Focus



APPENDIX 6: Summary of Gregor (2006) Theory Classification Scheme

Theory Type	Distinguishing Attributes
Analysis	Says what is. The Theory does not extend analysis and description. No causal relationships among phenomena are specified and no predictions are made.
Explanation	Says what is, how, why, when, and where. The theory provides explanations but does not aim to predict with any precision. There are no testable propositions.
Prediction	Says what is and what will be. The Theory provides predictions and has testable propositions but does not have well developed justificatory causal explanations.
Explanation and Prediction	Says what is, how, why, when, where, and what will be. Provides predictions and has both testable propositions and causal explanations.
Design and Action	Says how to do something. The theory gives explicit prescriptions (e.g. methods, techniques, principles of form and function for constructing an artefact).

APPENDIX 7: Summary of Llewelyn (2003) Theory Classification Scheme

Theory Type	Description
Level 1: Metaphor Theories	A metaphor is "a basic structural form of experience, through which human beings engage, organize and understand their world" (Morgan, 1983, p. 601). Metaphors can be used to express how organisations are understood or perceived.
Level 2: Differentiation Theories	Gregor (2006) posits that in the social science field, concepts are established through the use of metaphors and are further developed through differentiation or by contrasting concepts with other concepts. *Dualism* refers to the "opposing" nature of the key paired terms in differentiation theory. Categorisation can also be used to represent people's understanding of the social world
Level 3: Concepts Theories	Concepts can be part of other theories and can be theoretical tools themselves. They are used to observe and represent the world and act and work in it. During level three theorizing, new concepts are introduced in order to improve the discussion on the practical world and to provide new ways of thinking and acting in the world, and existing ones are refined
Level 4: Theorizing Settings	In this form of theorizing, the contextual influences on the individual, organizational, or social phenomena are defined. It reveals the social conditions under which practice emerges.
Level 5: Theorizing Structures	Theorizing structures is also referred to as grand theorizing. In this level of theorizing, the impersonal, large-scale, and enduring aspects of social life are explained, for instance, social institutions, culture, and class hierarchies. Theorizing structures operates in the "world of ideas" rather than in the "world of practice" and are derived from thinking about issues and relationships in an abstract way rather than empirically (Llewelyn, 2003).

APPENDIX 8: Classification of Literature Type

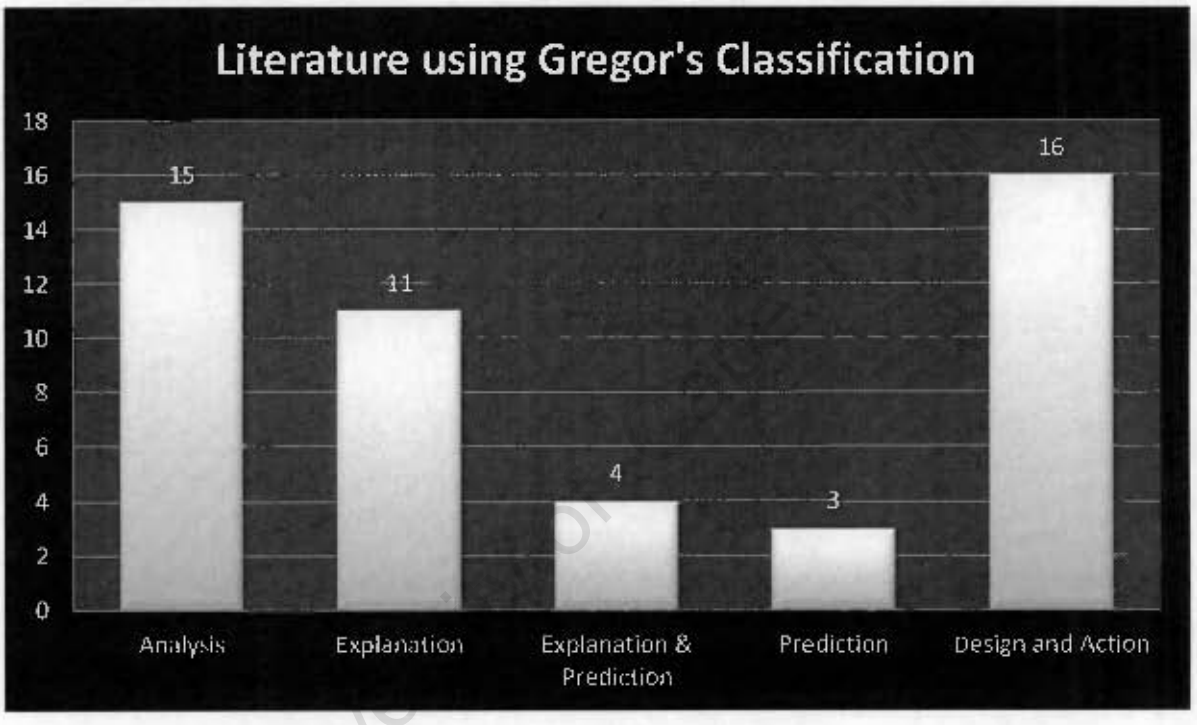
Paper Code	Paper Reference	Year	Industry	Academic
[S1]	Hazzan & Dubinsky, 2006	2006	1	
[S10]	Danait, 2005	2005	1	
[S13]	Sutherland, Schoonheim, Rustenburg, & Rijk, 2008	2008	1	
[S14]	Sutherland, Viktorov, Blount, & Puntikov, 2007	2007	1	
[S16]	Cannizzo, Marcionetti, & Moser, 2008	2008	1	
[S21]	Shrivastava & Date, 2010	2010	1	
[S22]	Williams & Stout, 2008	2008	1	
[S23]	Drummond & Unson, 2008	2008	1	
[S24]	Vax & Michaud, 2008	2008	1	
[S26]	Smits & Pshigoda, 2007	2007	1	
[S27]	Berczuk, 2007	2007	1	
[S28]	Summers, 2008	2008	1	
[S29]	Cottmeyer, 2008	2008	1	
[S30]	Therrien, 2008	2008	1	
[S33]	Jain, 2006	2006	1	
[S34]	Andrzeevski, 2007	2007	1	
[S36]	Young, & Terashima, 2008	2008	1	
[S37]	Robarts, 2008	2008	1	
[S38]	Uy & Ioannou, 2008	2008	1	
[S39]	Edwards, 2008	2008	1	
[S41]	Sheth, 2009	2009	1	
[S42]	Chubov, & Droujkov, 2007	2007	1	
[S45]	Miller	2008	1	
[S46]	Berteig	2007	1	
[S9]	Phalnikar, Deshpande, & Joshi, 2008	2008	1	
[S11]	Dullemond, van Gameren, & van Solingen, 2009	2009		1
[S12]	Bjørn & Ngwenyama, 2009	2009		1
[S15]	Paasivaara & Lassenius, 2006	2006		1
[S17]	Abbattista, Calefato, Gendarmi, & Lanubile, 2008	2008		1
[S18]	Whitworth & Biddle, 2007	2007		1
[S19]	Dorairaj, Noble, & Malik, 2010	2010		1
[S2]	Batra, 2009	2009		1
[S20]	Dorairaj, Noble J, & Malik, 2011	2011		1
[S25]	Layman, Williams, Damian, & Bures, 2006	2006		1
[S3]	Hossain, Bannerman, & Jeffery, 2011	2011		1
[S31]	Canfora, Cimitile, Di Lucca, & Visaggio, 2006	2006		1
[S32]	Yadav, Adya, Nath & Sridhar, 2007	2007		1
[S35]	Sison, & Yang, 2007	2007		1
[S4]	Holmström, Fitzgerald, Ågerfalk, & Conchúir, 2006	2006		1

[S40]	Urdangarin, Fernandes, Avritzer, & Paulish, 2008	2008	1
[S43]	Hossain, Ali Babar, & Verner, 2009	2009	1
[S44]	Schummer & Lukosch, 2008	2008	1
[S5]	Paasivaara, Durasiewicz, & Lassenius, 2008	2008	1
[S6]	Batra, Xia, VanderMeer, & Dutta, 2010	2010	1
[S7]	Green, Mazzuchi, & Sarkani, 2010	2010	1
[S8]	Ramesh, Cao, Mohan, & Xu, 2006	2006	1

APPENDIX 9: Literature Using Gregor's Classification

Paper Code	Year	Analysis	Explanation	Explanation & Prediction	Prediction	Design and Action
[S21]	2010	1				
[S33]	2006	1				
[S13]	2008	1				
[S22]	2008	1				
[S23]	2008	1				
[S24]	2008	1				
[S26]	2007	1				
[S28]	2008	1				
[S29]	2008	1				
[S36]	2008	1				
[S37]	2008	1				
[S38]	2008	1				
[S41]	2009	1				
[S42]	2007	1				
[S1]	2006	1				
[S9]	2008		1			1
[S3]	2011		1			
[S18]	2007		1			
[S39]	2008		1			
[S17]	2008		1			
[S19]	2010		1			
[S30]	2008		1			
[S43]	2009		1			
[S44]	2008		1			
[S45]	2008		1			
[S46]	2007		1			
[S6]	2010			1		1
[S7]	2010			1		1
[S40]	2008			1		1
[S35]	2007			1		
[S11]	2009				1	
[S12]	2009				1	
[S32]	2007				1	
[S25]	2006					1
[S27]	2007					1
[S2]	2009					1
[S4]	2006					1

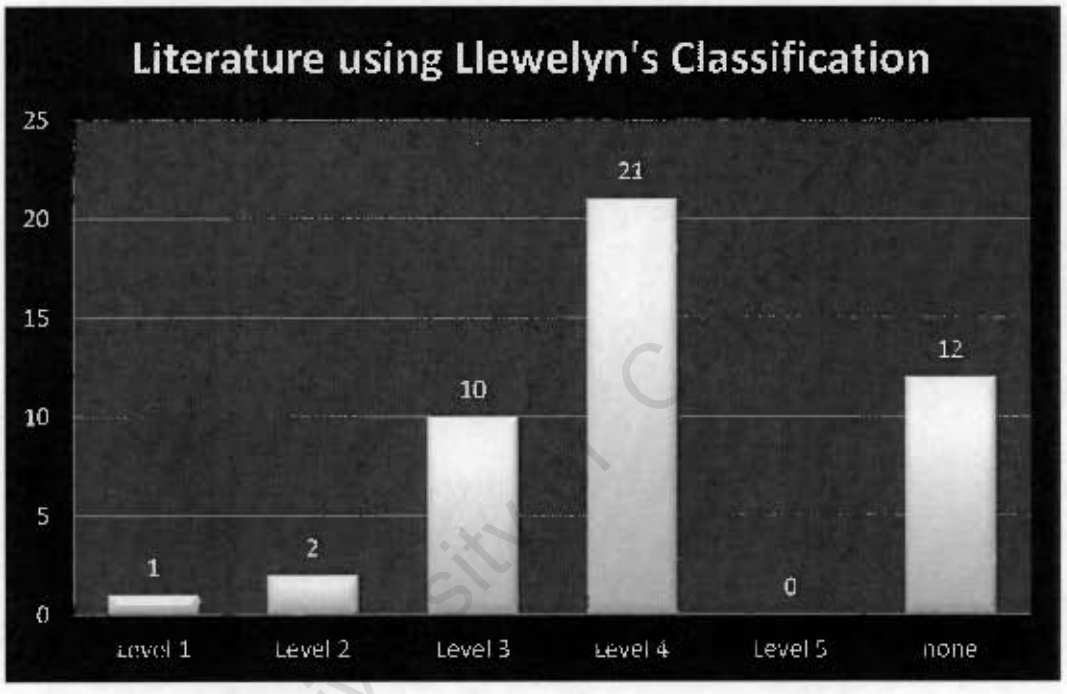
[S5]	2008	1
[S8]	2006	1
[S10]	2005	1
[S14]	2007	1
[S16]	2008	1
[S20]	2011	1
[S31]	2006	1
[S34]	2007	1
[S15]	2006	



APPENDIX 10: Literature Using Llewelyn's Classification

Paper Code	Year	Level 1	Level 2	Level 3	Level 4	Level 5	none
[S1]	2006	1					
[S2]	2009		1				
[S3]	2006		1				
[S4]	2011			1			
[S5]	2006			1			
[S6]	2010			1			
[S7]	2008			1			
[S8]	2008			1			
[S9]	2007			1			
[S10]	2008			1			
[S11]	2008			1			
[S12]	2006			1			
[S13]	2008			1			
[S14]	2006				1		
[S15]	2008				1		
[S16]	2010				1		
[S17]	2009				1		
[S18]	2009				1		
[S19]	2007				1		
[S20]	2010				1		
[S21]	2011				1		
[S22]	2010				1		
[S23]	2008				1		
[S24]	2008				1		
[S25]	2006				1		
[S26]	2007				1		
[S27]	2008				1		
[S28]	2007				1		
[S29]	2007				1		
[S30]	2008				1		
[S31]	2008				1		
[S32]	2009				1		
[S33]	2008				1		
[S34]	2007				1		
[S35]	2005						1
[S36]	2008						1
[S37]	2007						1
[S38]	2008						1

[S39]	2008						1
[S40]	2006						1
[S41]	2007						1
[S42]	2008						1
[S43]	2008						1
[S44]	2008						1
[S45]	2009						1
[S46]	2007						1
	Total	1	2	10	21	0	12



APPENDIX 11: Summary of Purposeful Sampling Dimensions Chosen for this Study

Theoretical construct	Theoretical construct description	Sampling Dimension	Case 1	Case 2
Fields	A domain or space in which GASD team members and their social positions are located (Bourdieu, 1990)	Number of software development sites	2	2
		Degree of cultural diversity at various sites	Team members from diverse cultural backgrounds, and belonging to diverse communities of practice	Team members from diverse cultural backgrounds, and belonging to diverse communities of practice
		Agile methodologies employed	Scrum	Scrum
		Past software development methodologies used by the team	Waterfall	Waterfall
		GASD format (e.g. offshore development, outsourcing)	Onshore/Offshore Software Development	Onshore/Offshore Software Development
Symbolic Capital	Symbolic capital pertains to prestige, honour, or the right to be listened to	Management structure (flat / hierarchical)	Flat	Hierarchical
		Level of experience of team members with software project	Team members' experience varied from being limited to high	Team members' experience varied from being limited to high
Economic Capital	Economic capital relates to the degree of command that an agent possesses over economic resources (e.g. cash or assets)	Team members / stakeholders monetary investment in the project	Team members' monetary investment in the project varied from being limited to high	Team members' monetary investment in the project varied from being limited to high
Cultural Capital	Cultural capital refers to knowledge, skills, education, and advantages that an agent possesses, thus giving him higher status in society	Level of experience of team members with Scrum	Team members' experience varied from being limited to high	Team members' experience varied from being limited to high
		Level experience of team members in various communities of practice within the software development industry	Team members' experience varied from being limited to high	Team members' experience varied from being limited to high

APPENDIX 12: Interview Guidelines

What is the context in which distributed agile projects are being implemented?

Background Information
• How old are you?
• How long have you been working for the company?
• How long have you been working in the team?
• What are your qualifications?
• What is your area of expertise?
• What you do and what is your role in the team?
• To what extent are you experienced with the use of Scrum?
• How would you describe the country X culture? What kind of people are they to work with?
• How would you describe your team culture?
The Organisation
• Tell me about your organisation
• How would you describe the management structure of the organisation
• How would you describe the organisational culture?
• How would you describe the "mind-set" behind software development in *country X*
The Team
• How was the team formed?
• How many people do you have in the team?
• How do you ensure that everyone is communicating as much as possible?
• How much input does the team give during the development process?
• Do you feel that you are listened to?
• Do you feel that your opinion is important and valued?
• To what extent are you able to communicate if you are not happy about something? Why?
• Have you ever met the team members at the other site? Who visited? How often?
• How are the team members being motivated?
Technology
• What communication tools do you use?
• How important are each of them?

- How frequently do you use each of them?
- Do you use a Scrum Tool? Which one?
- Tell me about the advantages of the Scrum Tool
- Tell me about the disadvantages of the Scrum Tool
- Which features of the Scrum tool do you use?
- Who chose the tool and why?
- Can you tell me how your development infrastructure is set up?

The Project

- Can you please tell me about the purpose of the project
- Why was the project initiated?

Why are particular distributed Scrum work practices adapted and followed? Why do some of these distributed Scrum work practices differ from traditional agile practices?

Scrum Implementation

- Why was Scrum chosen as a methodology?
- Have you been using Scrum since the beginning of the project?
- Describe the software development approach being used in the beginning?
- How has it changed? Why?
- Who decides on the changes? Why?

The Requirements

- Who decide on the requirements? Why?
- Who has the final say about the requirements? Why?
- Who prioritise the requirements? Why?
- How do you decide on task priority? Why
- How do you handle changes in requirements?
- What are the problems that happen during the user requirements elicitation?
- To what extent do requirements ever change in the middle of a sprint? Does the customer interfere in the middle of a sprint? Why?
- How can that process be improved?

The Distribution Environment

- How is the work distribution organised across the dispersed site?
- What are the challenges that you experienced while engaging with agile in a distributed environment?

- How do you overcome those challenges?
- How do you coordinate everything?
- Can you think of ways of improving what you are doing now?
- Do you feel any language barriers? How do you overcome them?

Agile

- How do you feel about following an agile approach which does not require so much documentation?
- Are you supposed to be CMM certified? How are the two reconciled?
- Agile prescribe that you should have a lot of interactions with your customers, to what extent does that happen here?
- Do you feel that interaction at your level? Explain
- Do you feel that you are influenced by the practices of the agile alliance? Explain
- Do you attend agile conferences? Read agile industry reports? Explain
- Do you adapt your practices based on what you read? Explain
- In what ways are you adapting the traditional Scrum work practices to fit the distributed setting?

The Process

- How long does one sprint last? Why?
- Does it happen that it lasts longer? Why?
- Tell me where is the analysis done, the development, do you do it both sides? Why did you decide to follow that approach?
- How is task allocation performed? Why?
- Who decide on how to improve on the software development process? (Retrospective meeting)
- When and by whom are decisions made? Why?
- How is testing performed in distributed environment?
- What kind of documentation do you use?
- How does deployment occur?
- How do you manage the task board?
- How do you take the distribution setting into consideration while estimating task duration?
- How does the retrospective happen?
- Do you use particular techniques during the retrospective meeting?
- Who attends the retrospective?
- To what extent is everyone comfortable raising concerns during the retrospective

meeting?

- How do you handle the bug fixing?

APPENDIX 13: Sample Field Note

Case Number	1
Date:	28th May 2010

General Comments:

Team info

XXX is the project manager. But he is currently also working as the Scrum Master. The original Scrum Master was YYY. But XXX took over the role since sprint 5. We are in sprint 9. The Product Owner is ZZZ. Casper publishes the stories, but he first confirms with Philip before publishing them.

Team members behaviour

People help each other out. Programmers move from their desks to their colleagues' desk to discuss and help.

At some point during the day, C2.R2 went round the table to see how each developer was progressing.

Security

The team seems very wary of security. They are unwilling to send snapshots of screen without approval from C2.R2.

Scrum Process

The Scrum board is at the client site. They used to send a snapshot of it to the team every day. But during the last couple of months, they were so busy in India that SA people did not send the Scrum board. I'm not sure about how a busy schedule can stop them from sending one email with a Scrum board as an attachment. I need to investigate that further.

The tasks are stored on a software called FogBugz.

Shridhar used to send a burn-down chart to the team daily to measure performance. But they are so busy that they don't do that anymore.

90% of design done in India.

Two planning meetings are performed. In the first one, they decide on the task complexity and assign the tasks. In the second one, they decide on the duration

In India, they don't keep track of the complexity but SA does.

High level tasks are decided by the project manager

Tasks breakdown are decided by developers

And in the sprint planning 2, they all give the estimates.

All these tasks might or might not be posted onto Fogbugz. If they are published on Fogbugz, it will be as a feature. If it's not published, they keep track of what they are doing in the daily Scrum meeting.

User stories come into play before breaking down the tasks. The Product Owner provides the user stories.

During sprint planning 1, user stories are not usually ready. Since this is a re-engineering project, the user stories are not available.

Office Space

The team moved to their new office today. I was waiting in the old office before realising that they had probably moved to the new office.

I can feel a change in atmosphere. The seating arrangement is so much better. I can see everyone and I feel part of the group even though I am not really one of them. I can only imagine what a difference it must make for all the team members.

Communication flows easily. Someone is singing. My impressing is that it will create better team cohesion.

C2.R2's desk is now far (as opposed to before) to the developers sitting place.

There are no phones on the desk of the developers.

Technology

Personal computers

Emails

Conference call

FogBugz

- FogBugz: Feature list, Bug List and all cases assigned to everybody.

- Milestone = Specifying when the task should be done

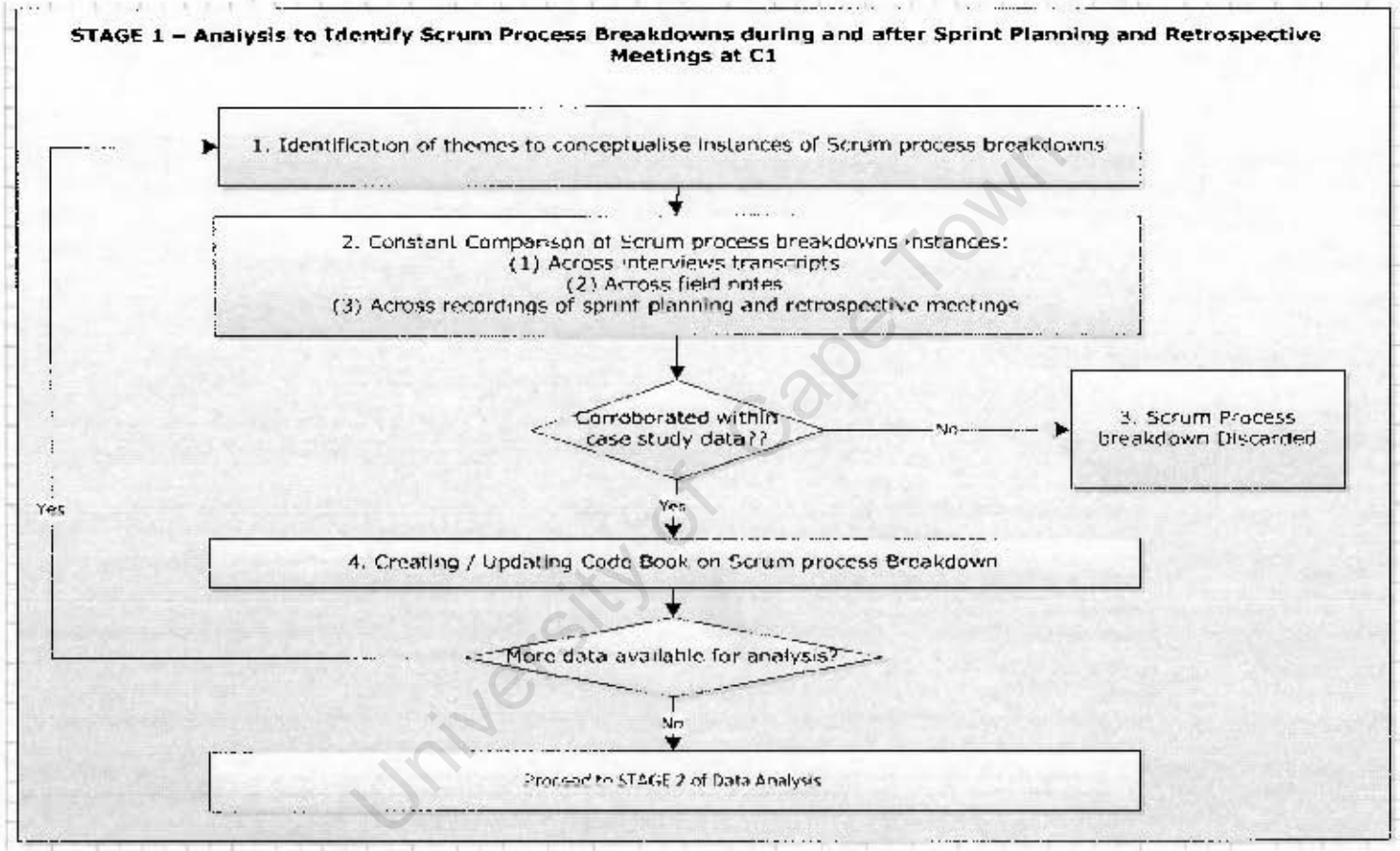
  E.g Undecided, Testing, Development, Sprint

- The milestone list is created by the product manager. ***Get more information about how the work is carried out***
- Automation category not used
- From FogBuz you can access the Wiki website
- Cases can be marked as duplicate if two people opened the same case
- Fogbuz can be used to manage the whole Scrum process. They use it without the Scrum board
- Everyone can create a task in the Fugbugz application.
- Tasks statuses are changed accordingly as they progress through the Scrum. A bit in a similar way as a task would be physically moved on the Scrum board.
- The main tools that they use are FugBugz, Wiki, and the Burndown chart
- The main screens of the FogBugz application has been sent by email

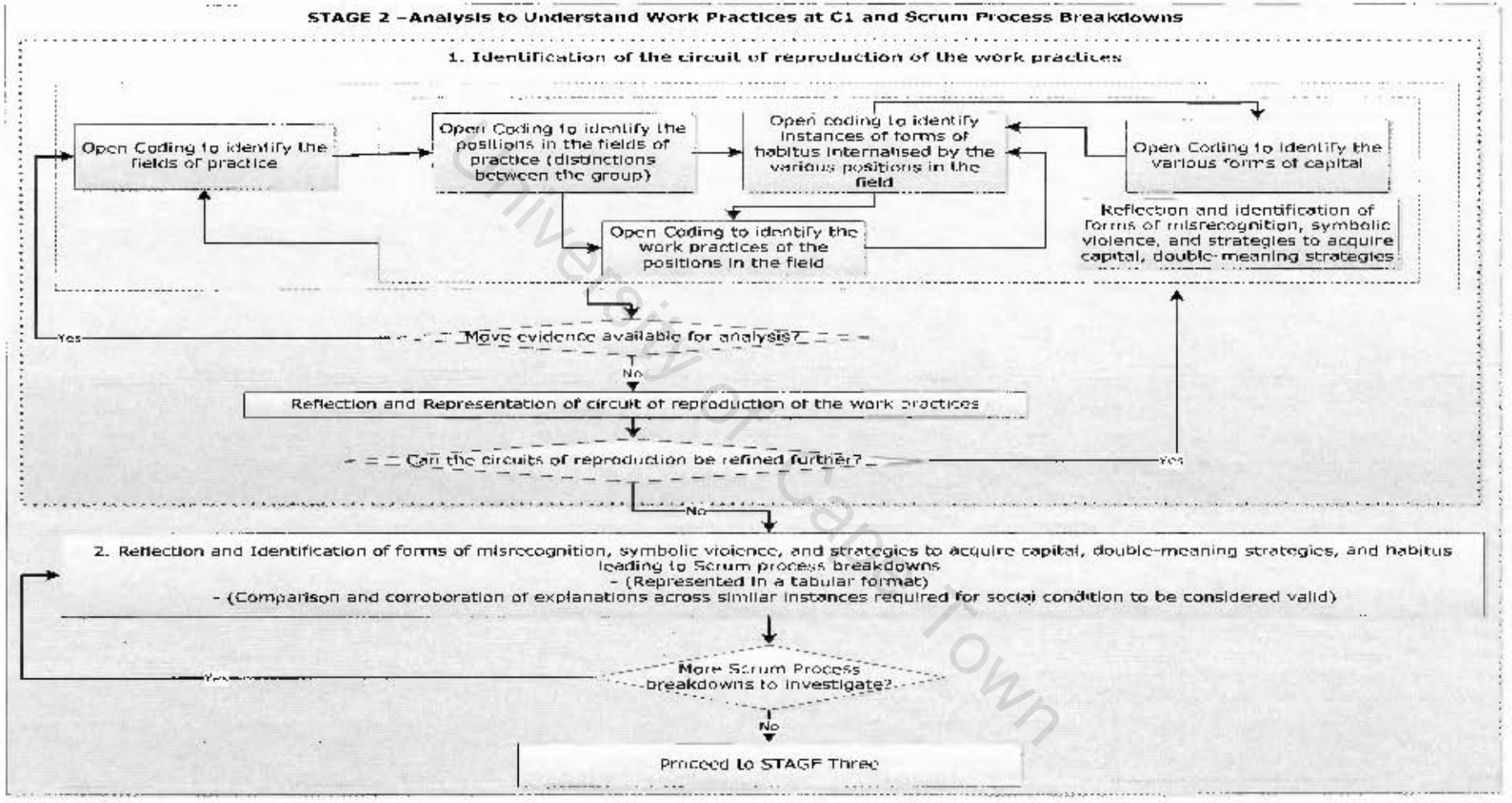
Wiki

- Under list, we have the different sprints info.
- Wiki is a content management program.
- The QA guys mostly use the Wiki as it is easier to track. But in FogBugz, they have to see all the cases.
- The Wiki can also have drawings of the screens with comments about the fields. The Product Owner is mainly the one who come up with the drawings, but sometimes the developer also provide the drawing.

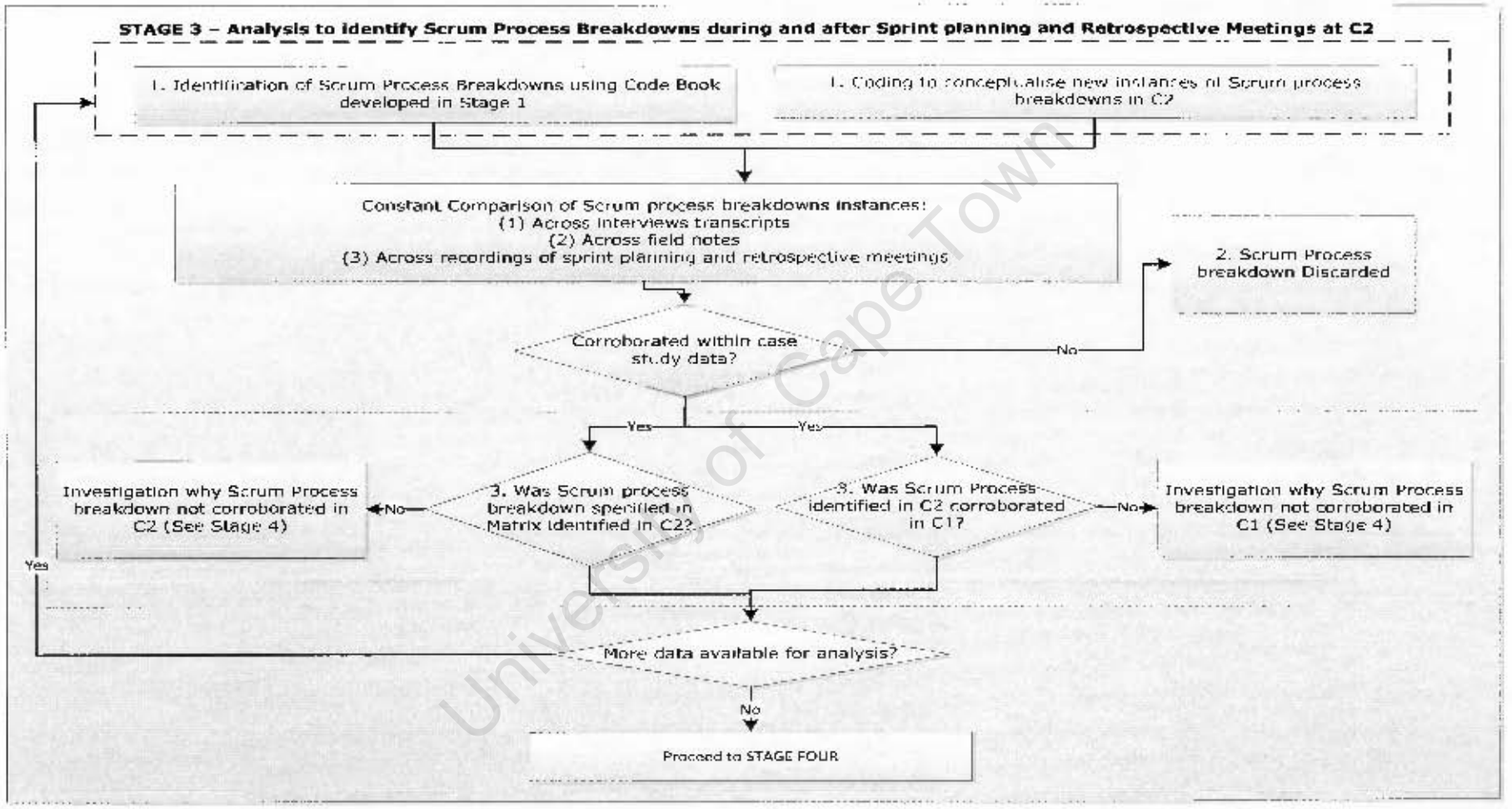
APPENDIX 14: Summary of Stage One of Data Analysis



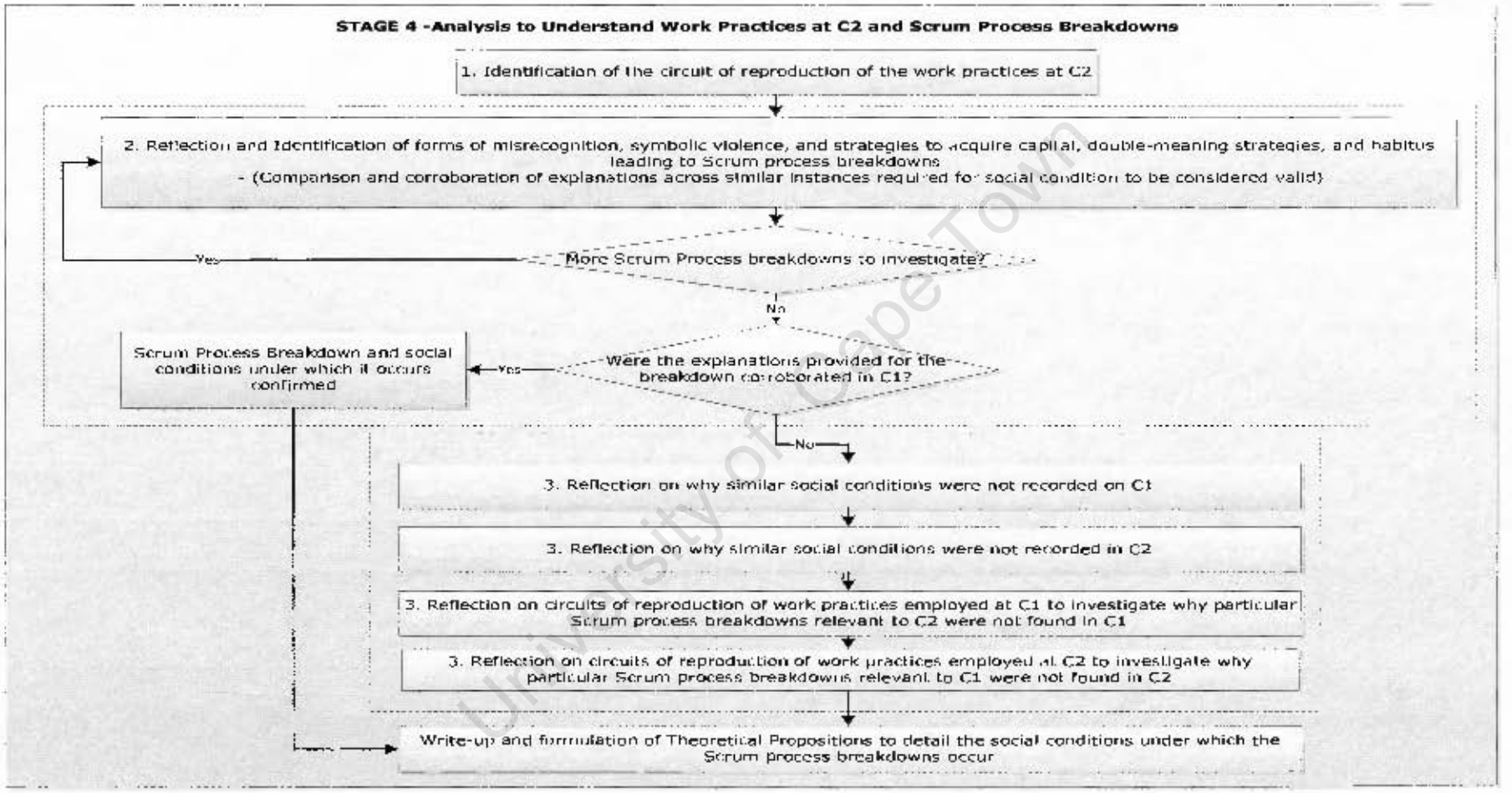
APPENDIX 15: Summary of Stage Two of Data Analysis



APPENDIX 16: Summary of Stage Three of Data Analysis



APPENDIX 17: Summary of Stage Four of Data Analysis



APPENDIX 18: Examples of Possible Scrum Process Breakdowns

Ideal Sprint Planning and Retrospective Meetings	Possible Scrum Process Breakdown Instances
Sprint Planning Meeting: Team should only select requirements which they can commit to complete during that sprint (Schuh, 2006)	Team is unable to complete all requirements that they committed to during the sprint planning meeting
Sprint Planning Meeting: The team and the Product Owner go through each item methodically to ensure that they understand the product requirement (Schwaber, 2009)	Lack of understanding of the product requirements from the team members leading to user requirements not being met
Sprint Planning Meeting: The entire team should engage in the estimation process and each team member should be given the chance to express their opinion on the task's complexity (Schwaber & Beedle, 2002)	Only some team members are able to express their opinion during the spring planning meeting's estimation process
	Team members are unable to engage in debates (Summers, 2008)
	Hard to establish good interaction between team members (Danait, 2005; Smits & Pshigoda, 2007)
	Lack of cultural understanding between team members (Drummond & Unson, 2008)
Sprint Planning Meeting: Once that the team and the Product Owner have agreed on the stories to be completed during the sprint, these stories are not supposed to change throughout the iteration (Schwaber, 2009)	User Requirements discussed and agreed upon during sprint planning meeting change in the middle of the sprint
Retrospective Meetings: Upon completing a retrospective meeting, the team should have identified a list of actionable set of measures to be followed to improve the Scrum process for the next sprint (Schwaber, 2009)	Sprint process not improved as per the recommendations of the team members
Retrospective Meetings:	Lack of understanding of term "self-organisation" (Batra et al., 2010

Teams should be self-organised	Formal relationship between vendor and customer (Batra et al., 2010)
Retrospective Meetings: Scrum Master should encourage the team to reflect on their current Scrum processes	Power imbalance (Therrien, 2008)

APPENDIX 19: Examples of Interview Statements for the 'Impromptu Changes to User Stories Mid-Sprint' Category

Interview Transcript	Category: Impromptu Changes to User Stories mid-sprint
Interview 2	The Product Owner and the CEO should ensure that they have a common understanding on the stories beforehand. What happened sometimes is that the CEO is very much involved with the project. In the middle of a sprint, requirements might change and he will reset the priorities of the tasks or change the user stories. This is a problem because it impacts on the programmers work load and all the other tasks
Interview 3	This is our second sprint now, and on the first sprint, as [the CEO] was away on multiple trips, only [the Product Owner] was available. And [the Product Owner] had created the priorities. And the actual description of the stories, he had actually created from an old discussion with [the CEO], so obviously, the communication hadn't been happening for a week or two. And when [the CEO] actually came back from the visits after a week or two, some of the stories changed dramatically.
Interview 4	*Does it happen that the story changes during one iteration?* It has happened. The previous one it did happen a couple of time
Interview 5	*Can you change priorities in the middle of an iteration?* I shouldn't ☺ I try not to, but if somebody tries says "it has to be done" and it becomes very apparent that it is, then we have to shift it. But I pay the price for that because we have to shift up the backlog. Because realistically speaking, one can only do so much.

APPENDIX 20: Scrum Process Breakdowns Code Book from C1

Occurrence	Category	Category Description
After Sprint Planning meetings	Impromptu Changes to User Stories mid-sprint	Changes in User Stories content and priorities mid-sprint
	Product Owner's low level of authority	Product owner's low level of authority in deciding on user stories content and priority in the team
During Sprint Planning meetings	Different perceptions about task urgency at the software development sites	Team members shared different perceptions about the urgency of completing the user stories (after mid-sprint changes to user stories)
During Sprint Planning meetings	Disagreements on software engineering practices	GASD team members disagreed on the nature of the tasks which the whole team should engage in during the sprint. For example, code refactoring to ensure reusability vs. writing of simple algorithms in order to complete the user stories as quickly as possible
During Sprint planning Meetings	Disagreements on estimation mechanisms	GASD team members disagreed on what's the best way to estimate the duration of a task
During Retrospective Meetings	Scrum process updates decisions not made by development team	Decisions on Scrum Process Update not made by the Development Team

APPENDIX 21: Summary of Categories and Themes for Scrum Process Breakdowns for C1

Sprint Planning Meetings Scrum Process Breakdowns			
	Sprint Interruptions		
		Impromptu Changes to User Stories' content and priorities	
			Change requests to user stories' content/description mid-sprint
			Change requests to user stories' priorities mid-sprint
	Collaboration Breakdowns		
		Product Owner's low level of authority	
			Product Owner's low level of authority pertaining to story elicitation
			Product Owner's low level of authority pertaining to story prioritisation
		Different perceptions about task urgency at the software development sites	
		Disagreements on suitability of software engineering practices	
			Disagreements about code optimisation tasks
			Disagreements about tasks duration
		Disagreements on estimation mechanisms	
			Preference for Task Estimation in hours
			Preference for Task Estimation in days

Retrospective Meetings Scrum process Breakdowns			
	Collaboration Breakdowns		
		Decisions on Scrum process updates not made by the development team	
			Scrum process update decisions made by COO
			Non implementation of team members ideas about Scrum process update

APPENDIX 22: Code Book for Theory of Practice Concepts

Theory of Practice Concepts	Explanation
Field	A domain or space in which GASD team members and their social positions are located (Bourdieu, 1990).
Position in the field	Result of the interaction between the demands and constraints in the field, the habitus of the GASD team members, and the amount of capital which they possess. GASD team members can also be in dominant, subordinate, or equivalent positions in the field (Bourdieu, 1990).
Habitus	Beliefs, values and dispositions that shape the conduct, thoughts, feelings, and judgments of the GASD team members. These can be identified through the practices which enact the habitus (Bourdieu & Wacquant, 1992).
Symbolic Capital	Symbolic capital is what gives some GASD team members more power over others. It relates to prestige, honour, or the right to be listened to (Bourdieu, 1993).
Cultural Capital	Refers to knowledge, skills, education, and advantages that a GASD team member possesses and which gives him or her higher status in society (Bourdieu, 1986).
Economic Capital	Economic capital relates to the degree of command that a GASD team member possesses over economic resources (cash or assets) (Bourdieu, 1986).
Practice	Relates to the actions that GASD team members engage in, or what they do. Practice is produced by habitus (Bourdieu & Wacquant, 1992). "Practices cannot be deduced either from the present conditions which may seem to have provoked them or from the past conditions which may have produced the habitus, the durable principal of their production. They can therefore only be accounted for by relating the social conditions in which the habitus that generated them was constituted, to the social conditions in which it is implemented" (Bourdieu, 1990).
Strategies	Elaborate or systematic plan of actions followed by distributed agile team members (Bourdieu, 1990).
Symbolic Violence	When team members with more capital than others in the field apply the power conferred by it against others who hold less. They try to impose their culture, ways of behaving, and / or thinking on others who hold less (Bourdieu, 1990; Jenkins 2002)
Double-Meaning Strategy	When GASD team members employ strategies and claim that they are working in the interest of the field, while in reality they are pursuing their own agenda to impose their own values in the field (Webb et al., 2002)
Misrecognition	When GASD team members who are victims of symbolic violence perceive this as legitimate and being according to the natural order of things (Webb et al., 2002)
Practice	Practice is produced because of the encounter between habitus and its disposition, as well as the constraints, demands and opportunities of the social field in which the actor is moving (Bourdieu & Wacquant, 1992).

APPENDIX 23: Summary of Categories and Themes for C1

Field: C1 GASD Team

Position in the field: Scrum Implementer and Driver

Meta - Category	Category	Concept
Habitus	It is important to reflect on Scrum and software development processes following by the team	Reflection on sprint length update
		Reflection on technology usage
		Reflection on estimation mechanism
Practices	Identification of challenges experienced during Sprints	ad hoc emailing of concerns
		Emailing of concerns at the end of the sprint
		Discussions during retrospective meetings
	Implementation of Scrum and Software Development Process Changes	Task estimation update
		Adoption of Scrum tool
		Adjustment of sprint length

Meta - Category	Category	Concept
Capital	Cultural Capital	Degree of Domain Application Knowledge
		Scrum Certification
		Experience in GASD Team
	Symbolic Capital	Job Title
		Scrum Implementer Role in Team
	Economic Capital	Degree of control of MKX application project

Field: C1 GASD Team

Position in the field: MKX Application Requirements Coordinators Work Practices

Meta - Category	Category	Concept
Habitus	It is important to ensure the usefulness of the MKX application functionalities for the end-users	Importance of MKX applications standard
		Importance of MKX applications' functionalities' standard
Practices	Final decisions on user stories made by CEO	Decisions on user stories priorities made by CEO
		Decisions on user stories content made by CEO
		Travel to gather requirements
	Testing and keeping up to date with MKX application deployment progress	Overlooking testing tasks
		Overlooking deployment tasks

Field: Software Developers
Position in the field: Technical Experts

Meta - Category	Category	Concept
Habitus	Communication within the team is important	
	Code Quality and Standard is important	Importance of Code Quality
		Importance of Code Standard
	Planning is Important	
	Technical Experts' skills is important	
Practices	Constant Communication is maintained	Open Communication Channels to avoid misunderstandings
		Daily Stand-up meetings
	Coding Strategy chosen by technical experts	

	Tasks allocated to technical experts accoding to skills	
	Code Reviews	
	Structured process followed during sprint	Structured process to handle bug fixing
		Structured process to handle task dependencies
		Working on high priority stories first
	Practices to improve skills	Allocation of tasks to team members to improve skills
		Travel to promote knowledge sharing across team
		Leaning from experts from the wider community

Meta - Category	Category	Concept
Capital	Cultural Capital	Degree of expertise on software development tasks for MKX application
	Symbolic Capital	Having a useful area of expertise

Field: Sao Paulo Team

Position in the field: Brazilian Software Developers

Meta - Category	Category	Concept
Habitus	Importance of a Relaxed Work Atmosphere	Importance of being able to chose work style
		Importance of flexible work hours
	Importance of Precision during Coding	
	Importance of Communication	Defensive about point of view
		limited diplomacy while communicating
Practices	Team is self-managed	
	Always pushing for code optimisation	

	Constant communication	constant communication but abrupt style
		communication through multiple channels
		frequent communication

Meta - Category	Category	Concept
Capital	Cultural Capital	Knowledge of Programming Language
		Expertise in Software Development Tasks
	Symbolic Capital	Seniority Level

Field: Cape Town Team
Position in the field: Cape Town Software Developers

Meta - Category	Category	Concept
Habitus	Importance of being output oriented	
Practices	Responding to emails and phone calls outside working hours	
	Volunteering to work extra hours if required	

Meta - Category	Category	Concept
Capital	Cultural Capital	Degree of expertise on software development tasks for MKX application
		Degree of expertise on MKX software application architecture
	Symbolic Capital	Years of experience on MKX application

Field: Brazilian Software Developers field
Position in the field: PHP Experts

Meta - Category	Category	Concept
Habitus	Technical aspect of coding is important	
Practices	Attending PHP Conferences to learn about PHP	
	Communicating desired PHP Code standards by email	

Meta - Category	Category	Concept
Capital	Cultural Capital	PHP Software Development Skills
	Symbolic Capital	Degree of Influence within the overall PHP community

APPENDIX 24: Summary of Scrum Process Breakdowns and Social Conditions Leading to them in C1

Scrum process breakdown during and after sprint planning meetings	**Social conditions leading to breakdowns**	**Empirical observation in C1**
Impromptu changes to user stories' content and priorities	• Dominant team members seeking to acquire more capital	• CEO seeking to retain control over the project
	• Software developers forced to engage in practices that go against what their habitus predispose them to.	• Conflict accentuated when software developers were forced to deal with changes in user-stories, a work practice which was against their habitus predisposing them to plan ahead.
Product Owner's low level of authority	• Lack of capital in comparison to other team members	• Product Owner's low level of authority in comparison to CEO
Different perceptions about task urgency at the software development sites	• Team members Habitus predispose them to focus on conflicting tasks	• Software developers belonging to separate fields and internalised conflicting habitus
Disagreements on suitability of software engineering practices	• Team members Habitus predispose them to focus on conflicting tasks	• Software developers belonged to separate fields, and had internalised divergent and conflicting habitus
	• Conflicting Strategies to acquire capital	• Software developers belonged to separate fields and competed for conflicting capital
Disagreements on estimation mechanisms	• Software developers forced to engage in practices that go against what their habitus predispose them to.	• Brazilian software developers forced to work in a controlled environment when they also belonged to a sub-field within which they had internalised habitus predisposing them to value a relaxed work atmosphere
	• Lack of capital in comparison to other team members	• Brazilian software developers could not negotiate against the

		decisions of the COO because of a lack of capital

Summary of Scrum Process Breakdowns during and after sprint planning meetings and social conditions under which they occur

Scrum process breakdown during and after retrospective meetings	**Social conditions leading to breakdowns**	**Empirical observation in C1**
Decisions on Scrum Process Update not made by the Development Team	• Lack of capital in comparison to other team members	Software Developers at C1 lacked capital in comparison to C1.R1 who owned more capital and could implement views on how to update the Scrum process

Summary of Scrum Process Breakdowns during and after Retrospective meetings and social conditions under which they occur

APPENDIX 25: Summary of Categories and Themes for Scrum Process Breakdown for C2

Sprint Planning Meetings Scrum Process Breakdowns			
	Sprint Interruptions		
		Impromptu Changes to User Stories' content and priorities	
			Change requests to user stories' content/description mid-sprint
			Change requests to user stories' priorities mid-sprint
	Collaboration Breakdowns		
		Number of User Stories to be completed during the Sprint is imposed on the team	
			Large number of requirements requests from customer
			Team's limited ability to negotiate against large number of requirements
			Sprint backlog decision imposed on team
			Team's limited ability to convey task complexity to customers
		Disagreements on estimation mechanisms	
			Preference for Task Estimation in hours
			Preference for Task Estimation in days

		Low level of communication openness during meetings	
			team members lack of communication during international meetings
Retrospective Meetings Scrum process Breakdowns			opinions not easily communication
			open communication during internal meetings
			debates during internal meetings
	Collaboration Breakdowns		
		Decisions on Scrum process updates not made by the development team	
			Scrum process update decisions made by COO
			Non implementation of team members ideas about Scrum process update
		Selective Invitation to Retrospective meetings	
			Indian team members non involvement in retrospective meetings
			filtering of communication via project communication in india

APPENDIX 26: Summary of Categories and Themes for C2

Field: C2 GASD Team

Position in the field: Project Managers

Meta - Category	Category	Concept
Habitus	Tracking Performance and Progress is Important	Importance of having a detailed view of effort being made
		Importance of tracking project completion progress
	Hierarchical organisational structure is important	
	Producing highly qualified team members is important	
Practices	Implementation of Performance and Progress Tracking mechanisms	Review meetings to discuss performance and code quality
		Tracking of team velocity
		Using of technology to track who is doing what
		Monitoring the execution of customers' requests
		The use of an onsite coordinator
	Formal communication mechanisms between software developers, Project Managers and customers	Requests for official emails from software developers to express grievances
		Retrospective meetings between customers and management only
	Official and on the job training for team members	Organising training sessions on agile and Scrum
		Organising training sessions on the Colin application project

Meta - Category	Category	Concept
Capital	Cultural Capital	Number of years of experience in project team
		Experience in agile software development
		Current project experience

	Symbolic Capital	Job Title
	Economic Capita	Ownership of project

Field: C2 GASD Team
Position in the field: Testers

Meta - Category	Category	Concept
Habitus	Up-to-date and precise requirements are valued	
Practices	Constant requests for software developers and customers to provide up to date information on requirements	Constant communication to obtain clarifications on requirements
		Participating in daily stand-ups to understand requirements
		Pushing for documentation requirements to be kept up-to-date

Field: Pune Team
Position in the field: GASD Software Developers

Meta - Category	Category	Concept
Habitus	Productivity is important	"Doers" mentality
		Importance of satisfying the customer
	Adherence to standards is important	Importance of abiding by agile standards
		Importance of abiding by coding standards
		Importance of code quality
	Collaboration is important	
	Planning in important	
Practices	Product Demo	Showcasing achievements throughout the sprints
		Obtaining feedback from customers
		Identifying opportunities for improvements

	Work extra hours	
	Impediments notification	Informing the entire team that work is on hold
		Mechanisms to obtain fast feedback on information requested
	Code Reviews	
	Constant communication to promote group collaboration	Discussions on task estimation prior to and during the sprint planning meetings
		Phone discussion to finalise estimation process
		Group discussion on how best to complete a task in case of uncertainty
	Architecture and technology planning	Attention paid to adequacy of architecture
		Attention paid to adequacy of technology employed
	Sprint Planning Meetings	Sprint planning meeting 1
		Sprint Planning meeting 2

Meta - Category	**Category**	**Concept**
Capital	Cultural Capital	Degree of technical knowledge on the software development tasks for Colin Application
		Academic Qualification
	Symbolic Capital	Seniority Level

Field: Durban Team

Position in the field: Customers

Meta - Category	Category	Concept
Habitus	Retaining control over the project is important	Importance of daily clarity on project work progress
		Importance of daily clarity on software developers time management
	Project deadlines and quality are important	Software developers not allowed to miss deadlines
	□	Importance of maintaining code quality
Practices	Deciding on user stories and priorities	Deciding on requirements based on demands of end users
		Deciding on requirements priorities
	Approval of new team members prior to them joining the team	
	Keeping database and web servers onshore	
	Decisions on methodological processes followed by the team	Burn-down chart and task board maintained in onshore
		Bypassing Scrum work practices to focus on completing project tasks
		Choosing Scrum as a methodology
		Wiki update
	Weekly retrospective meetings in case of project urgency	

Meta - Category	Category	Concept
Capital	Cultural Capital	Degree of expertise on the Business Logic of the Colin Application
	Symbolic Capital	Job Title
	Economic Capital	Degree of control on the Casper Application Project

Field: Durban Team

Position in the field: Onsite Coordinators' Work Practices Circuit of Reproduction

Meta - Category	Category	Concept
Habitus	Providing project visibility to the customer is important	
Practices	Informing customers of offshore team's views	Liaising with team members to obtain status update
		Informing customers of team members' status update and requests
	Managing customer's requirements requests	Assessing technical feasibility of customers' requests
		Obtaining precise requirements from the customer

Field: Sanbi (C2 Organisation)

Position in the field: Project Managers

Meta - Category	Category	Concept
Habitus	Abiding by standards is important	Importance of abiding by agile standards
		Importance of abiding by design and coding standards
		Importance of abiding by industry standards
		Importance of meeting the demands of the market
	Meeting the customers' demands is important	Importance of meeting the customers' agile needs
		Importance of satisfying the customers' demands
Practices	Adapting Scrum work practices to industry best practices	Merging of Agile and CMMI practices
		Following IT industry experts and trends
		Careful selection of agile trainers
	Negotiating with customers to understand and implement their agile needs	

APPENDIX 27: Decision Tables used to Formulate Theoretical Propositions

Initial Version of Decision Table

Conditions	P1	P11	P7	P6	P2	P3	P4	P8	P10	P12	P5
Retention and Acquisition of economic Capital	1	1	0	0	0	0	0	0	0	0	0
Different orientations towards capital acquisition	0	0	1	0	0	0	0	0	0	0	0
Limited Symbolic Capital	0	0	0	0	0	0	1	1	1	0	0
Limited Cultural Capital	0	0	0	0	0	0	1	1	1	0	0
Limited Economic Capital	0	0	0	0	0	0	0	0	0	1	0
Valuing the importance of planning	0	0	0	0	1	0	0	0	0	0	0
sharing different beliefs and values because they inhabit multiple fields	0	0	0	1	0	1	0	0	0	0	1
Outcome											
high number of request for Impromptu requirements changes	1	0	0	0	0	0	0	0	0	0	0
high resistance to impromptu requirements changes	0	0	0	0	1	0	0	0	0	0	0
high number of requirements are imposed on the SDTM	0	0	0	0	0	1	1	0	0	0	0
limited ability to decide on requirements content and priorities	0	0	0	0	0	0	0	0	0	1	0
few ideas from SDTM implemented on software development process improvement	0	0	0	0	0	0	0	0	1	0	0
Divergent reaction to urgency	0	0	0	0	0	0	0	0	0	0	1
frequent disagreement on software engineering practices	0	0	1	1	0	0	0	0	0	0	0
limited open communication	0	0	0	0	0	0	0	1	0	0	0
Customers Control access to SDTM resources and opportunities	0	1	0	0	0	0	0	0	0	0	0

Final Versions of Decision Tables

Conditions	P1	P11	P7
Retention and Acquisition of economic Capital	1	1	0
Different orientations towards capital acquisition	0	0	1
Outcome			
high number of request for Impromptu requirements changes	1	0	0
Customers Control access to SDTM resources and opportunities	0	1	0
frequent disagreement on software engineering practices	0	0	1

- TP1 - When SDT stakeholders seek to acquire economic capital, the SDTM will face high number of requests for impromptu requirements changes AND will have limited access to resources and opportunities
    - TP1a – When SDT stakeholders seek to acquire economic capital, the SDTM will face high number of requests for impromptu requirements changes
    - TP1b – When SDT stakeholders seek to acquire economic capital, the SDTM will have limited access to resources and opportunities
- TP2 - When SDTM have different orientation towards capital acquisition, there could be frequent disagreements on software engineering practices

Conditions	P4	P8	P10	P12
Limited Symbolic Capital	1	1	1	0
Limited Cultural Capital	1	1	1	0
Limited Economic Capital	0	0	0	1
Outcome				
high number of requirements are imposed on the SDTM	1	0	0	0
limited ability to decide on requirements content and priorities	0	0	0	1
limited open communication	0	1	0	0
few ideas from SDTM implemented on software development process improvement	0	0	1	0

- TP3 - When the SDTM have limited capital, they will have limited ability to decide on the requirements number, content, and priorities AND will be less open during communication AND few of their ideas on how to update the process will be implemented
    - TP3a - When SDTM have limited symbolic or cultural capital, a high number of requirements will be imposed on the team
    - TP3b - When the Product Owner has limited cultural and economic capital, they will have limited ability to decide on the requirements content and priorities
    - TP3c - When the SDTM have limited symbolic or cultural capital, they will be less open during communication
    - TP3d - When the SDTM have limited symbolic or cultural capital, few of their ideas on how to update the software development process will be implemented

Conditions	P6	P2	P3	P5
Valuing the importance of planning	0	1	0	0
sharing different beliefs and values because they inhabit multiple fields	1	0	1	1
Outcome				
frequent disagreement on software engineering practices	1	0	0	0
high resistance to impromptu requirements changes	0	1	0	0
high number of requirements are imposed on the SDTM	0	0	1	0
Divergent reaction to urgency	0	0	0	1

- TP4 - When SDTM share different beliefs and values because they inhabit multiple fields, there will frequent disagreements on software engineering practices AND divergent reactions to urgency AND a high number of requirements will be imposed onto them
    - TP4a – When SDTM share different beliefs and values because they inhabit multiple fields, there will be frequent disagreements on the suitability of software engineering practices
    - TP4b – When SDTM share different beliefs and values because they inhabit multiple fields, they will have divergent reactions to urgency
    - TP4c - When the Scrum Master shares different beliefs and values because he/she inhabits multiple fields, a high number requirements will be imposed on the team
- TP5 - When SDTM value the importance of planning, there will be high resistance to impromptu requirements changes

APPENDIX 28: Interview Consent Form

UNIVERSITY OF CAPE TOWN

Department of Information Systems

Leslie Commerce Building
Engineering Mall. Upper Campus
OR Private Bag. Rondebosch 77001
Cape Town
Tel: 650-2261
Fax No: (021) 650-2280

INTERVIEW PARTICIPATION CONSENT FORM

As requirements for completing a PhD at the Department of Information Systems in the Faculty of Commerce at the University of Cape Town the researcher, Maureen TANNER, is performing a study entitled "*Understanding the social and contextual influences on Distributed Agile Software Development: A Theory of Practice Perspective*".

The main research objectives of this study are to:

- To reveal the social and contextual factors influencing and shaping the manner in which agile practices are being followed in the GSD context
- To understand why particular distributed agile practices are being devised and followed
- To understand why do some of these distributed agile practices differ from what is generally prescribed by traditional agile standards
-

An issue that is of utmost importance to the researcher, the department, the faculty and the University of Cape Town at large is research ethics. Consequently, the researcher guarantees the confidentiality and anonymity of the details and comments you provide, which will strictly be used for the sole purpose of the aforementioned research report. Furthermore, your participation in this study is entirely voluntary. You may choose to be excluded from the study at any point in time without incurring any adverse consequences. If you so choose to be involved with this research project, please sign the consent form below.

PARTICIPANT CONSENT FORM

By signing this participant consent form, you are agreeing to participate in a research project entitled "*UNDERSTANDING THE SOCIAL AND CONTEXTUAL INFLUENCES ON DISTRIBUTED AGILE SOFTWARE DEVELOPMENT: A THEORY OF PRACTICE PERSPECTIVE*" conducted by Maureen TANNER as requirement for the course entitled PhD in Information Systems. The researcher guarantees the confidentiality and anonymity of the details and comments you provide, which will strictly be used for the sole purpose of the aforementioned research report. Should you wish to contact the researcher for any reasons whatsoever, please do not hesitate to email her at M.tanner@uct.ac.za or call her on +2778 620 9418

Contact Name: _____
Company Name: _____
Signature: _____
Date: _____

APPENDIX 29: Summary of Scrum Process Breakdown Occurring During and After Sprint Planning Meetings and their Social Conditions in C1 and C2

Scrum Process Breakdown during and after sprint planning meetings	**C1**	**C2**	**Conditions and Factors leading to Breakdowns**	**Conditions and Factors in C1?**	**Conditions and Factors in C2?**
Sprint Interruptions Impromptu changes to user stories' content and priorities	Yes	Yes	• Dominant team members seeking to acquire more capital	Yes – CEO seeking to retain control over the project	Yes – Customers seeking to retain control over the project
			• Lack of capital in comparison to other team members	Yes – Software Developers lacked capital and their attempts in stopping the CEO from changing stories mid-sprint failed	No – no attempt to stop the changes
			• Software developers forced to engage in practices that go against what their habitus predispose them to.	Yes – conflict accentuated when software developers were forced to deal with changes in user-stories, a work practice which was against their habitus predisposing them to plan ahead.	No – less conflict as software developers were not forced to behave differently to what their habitus predisposed them to. Strong habitus of having to please the customer, hence accepting the user stories changes

Collaboration Breakdowns Product Owner's low level of authority	Yes	No	•	Lack of capital in comparison to other team members	Yes – Product Owner's low level of authority in comparison to CEO	No – Product Owner belongs to customer's field, and has enough economic capital specify user stories content and priorities, hence no breakdown
Collaboration Breakdowns Different perceptions about task urgency at the software development sites	Yes	No	•	Team members Habitus predispose them to focus on conflicting tasks	Yes – software developers belonging to separate fields and internalised conflicting habitus	No software developers' habitus harmonised and they all belonged to one field
Collaboration Breakdowns Disagreements software engineering practices	Yes	No	•	Team members Habitus predispose them to focus on conflicting tasks	Yes – Software developers belonged to separate fields, and had internalised divergent and conflicting habitus	No – Software developers all belonged to one field and were not subjected to conflicting habitus
			•	Conflicting Strategies to acquire capital	Yes – software developers belonged to separate fields and competed for conflicting capital	No – software developers all competed for the same form of habitus
Collaboration Breakdowns Number of User Stories to be completed during the Sprint is imposed on the software developers	No	Yes	•	Team members Habitus predispose them to focus on conflicting tasks	No – Scrum Master and software developers belonged to same field and internalised same habitus	Yes – Scrum Master belonged to different field to software developers and internalised habitus predisposing him to focus of tasks conflicting with his role as Scrum Master
			•	Lack of capital in comparison to other team members	No – Scrum Master and software developers belonged to same field and competed for same capital	Yes Software Developers lacked capital and could not successfully negotiate the number of user stories which they would be comfortable to complete during the sprint

Collaboration Breakdowns Low level of communication openness	No	Yes	• Lack of capital in comparison to other team members	No – Team members had enough capital in the main field to openly expressed themselves and be listened to	Yes – Some team members had more capital in their sub-fields in comparison to the main field, and were thus more open there
Collaboration Breakdowns Disagreements on estimation mechanisms	Yes	No	• Software developers forced to engage in practices that go against what their habitus predispose them to.	Yes – Brazilian software developers forced to work in a controlled environment when they also belonged to a sub-field within which they had internalised habitus predisposing them to value a relaxed work atmosphere	No – Software developers were not forced to engage in practices conflicting with what their habitus predisposed them to

APPENDIX 30: Summary of Scrum Process Breakdown Occurring During and After Retrospective Meetings and their Social Conditions in C1 and C2

Scrum Process Breakdown during and after Retrospective meetings	**C1**	**C2**	**Conditions and Factors leading to Breakdowns**	**Conditions and Factors in C1?**	**Conditions and Factors in C2?**
Collaboration Breakdowns Decisions on Scrum Process Update not made by the Development Team	Yes	Yes	• Lack of capital in comparison to other team members	Software Developers at C1 lacked capital in comparison to C1.R1 who owned more capital and could implement views on how to update the Scrum process	Software Developers at C2 lacked capital in comparison to the customers who retained control over the Scrum process by deciding on changes to be implemented
Collaboration Breakdowns Selective Invitation to Retrospective Meetings	No	Yes	• Dominant team members seeking to prevent other team members from acquiring capital	No – Software developers had enough capital to attend the meetings	Yes - C2, customers employed this strategy to maintain their dominant position in the C2 GASD Team field and the Durban Team field, while management supported this practice in order to maintain their dominant position in the C2 GASD Team field

APPENDIX 31: Summary of Scrum Process Breakdown and their Social Conditions

Scrum Process Breakdown	*Social Conditions leading to Breakdown*
Impromptu change requests in user stories' content and priorities	• Acquisition of economic capital (TP1a) • Oriented to value the importance of planning (TP5)
Product Owner's low level of authority	• Limited capital in the joint field (TP3b)
Different perceptions about task urgency at the software development sites	• Different beliefs and values because of multiple fields (TP4b)
Disagreements on software engineering practices	• Different beliefs and values because of multiple fields (TP4a) • Divergent orientations towards capital acquisition (TP2)
Number of User Stories to be completed during the Sprint is imposed on the team	• Different beliefs and values because of multiple fields (TP4c) • Limited capital in the joint field (TP3a)
Low level of communication openness	Limited capital in the joint field (TP3c)
Disagreements on estimation mechanisms	• Team members Habitus predispose them to focus on conflicting tasks (TP4a)
Selective invitation to retrospective meetings	• The desire to acquire of more economic capital (TP1b)
Decisions on Scrum process update not made by the development team during retrospective meetings	• GASD project stakeholders' low level of capital in the joint field (TP3d)

APPENDIX 32: Proposed Best Practices while Implementing Agile and Scrum work practices in GASD

Agile Principles	Social Challenges	Proposed improved Mitigating Practices (Best Practices)
Motivated team members	- Lack of control on motivation at diverse site (Batra, 2009; Batra et al., 2010) - No personal relationship to build on (explanation for lack of motivation) (Phalnikar et al., 2008)	**TP4b** Scrum Master should strive to understand the various beliefs and values of the team members from all the dispersed sites (e.g. why they react in certain ways when faced with urgent situation) as a means to motivate them and leverage productivity.
Self-organising teams	- Lack of understanding of term "self-organisation" (Batra et al., 2010) - Cultural disparities (Batra, 2009) - Formal relationship between vendor and customer (Batra et al., 2010)	**TP3** Distributed Scrum team members with limited ability to participate in the self-organising aspect of the team should reflect of the types of capital recognised in the distributed Scrum team and strive to acquire more of those for their opinion to be recognised.
Working software	- Clashing cultural values (Dullemond et al.,	**TP4a**

	2009; Berteig, 2007)	Organise the work such that team members with similar beliefs and values collaborate more closely together on certain parts of the project so that software engineering practices are better aligned.
Welcome changes in requirements	- Inability to negotiate contract (not explained why) (Batra et al., 2010)	**TP3** Distributed Scrum team members with limited ability to negotiate during the sprint planning meetings reflect of the types of capital recognised in the distributed Scrum team and strive to acquire more of those for their opinion to be recognised
Reflection	- Cultural clashes (Batra, 2009) - Power imbalance (Therrien, 2008)	**TP3** In order to effectively empower offshore team members to fully participate in the decision making process, they should be given the opportunity to acquire enough recognised capital within the GASD team. Alternatively, measures should be taken so that their current capital is valued by all onshore team members.