Expertise Coordination for Agile Software Development Projects

by

Mawarny Md. Rejab

A Thesis
submitted to Victoria University of Wellington
in fulfilment of the
requirements for the degree of
Doctor of Philosophy
in Software Engineering.

Victoria University of Wellington
2017

Abstract

Agile software development projects rely on the diversity of team members' expertise. This expertise, however, is not adequate on its own: it is important to leverage available expertise through expertise coordination. Expertise coordination requires team members to rely on each other for recognizing who has particular expertise, when and where they are needed, and how to access the expertise effectively. Agile teams also need to rely on outside expertise such as user experience designers, architects, and database administrators. This thesis presents a theory of expertise coordination in Agile Software Development projects. We employed semi-structured interviews, observations, and document analysis in a Grounded Theory study involving 48 Agile practitioners and external specialists. This study discovered three main categories of expertise coordination: processes of expertise coordination, strategies of managing external expertise, and management roles in supporting expertise coordination. The theory provides a new insight into how Agile teams coordinate internal and external expertise, how they utilize external specialists and outsourcers' expertise, and how management can support expertise coordination.

Acknowledgments

I would like to express my deepest gratitude to my supervisors Prof. James Noble and Dr. Stuart Marshall for the continuous support of my PhD study and journey, for their diligence in reading and commenting on various versions of this thesis, motivation, immense knowledge, and financial support. Their guidance assisted me in conducting this study and writing this thesis.

Thank you so much to my lovely and supportive husband, Afzainizam Ab. Rahim, who accompanied me from Malaysia to New Zealand, for making my dream come true. My lovely kids, Mya Aleeyah Afzainizam and Emir Rayyan Afzainizam, who are motivations for me to keep writing and finishing this thesis. I strongly believe that without the prayers and blessing of my parents, I would not have been able to complete this thesis.

I wish to express my appreciation to Universiti Utara Malaysia and the Ministry of Higher Education Malaysia for a PhD scholarship. Thank you so much to Victoria University of Wellington for travel grants to Florida, US, and Rome, Italy.

I wish to thank all participants involved in this study who contributed their time to share their experiences, ideas and thoughts as inputs for this study. Thank you so much to Dr. George Allan for supporting me at the beginning of my PhD journey. Many thanks to all my lovely true friends in New Zealand and Malaysia for encouraging and supporting me through this entire PhD journey.

Contents

List of Tables

List of Figures

1

Introduction

Agile Software Development projects emphasize effective teamwork by concentrating on people. People are the most important factor in determining the success of Agile Software Development projects [39, 59]. Agile methods are designed to capitalize on every individual's expertise, but to produce a quality software product, the presence of expertise is not sufficient [54]; it is important to leverage the available expertise through expertise coordination.

Expertise coordination refers to how team members depend on each other to manage and utilize their expertise. Expertise coordination requires a team to recognize who has particular expertise, when and where that expertise is needed, and how to access the expertise effectively in a timely manner [54].

A number of researchers have studied coordination in the Agile Software Development context [146, 149, 167, 169, 194, 205]. Most studies emphasize determining the Agile practices and artefacts that have a significant influence in achieving coordination. A few empirical studies, however, focus on developing a coordination theory. Strode and Huff [205] propose a

theoretical model of coordination in Agile Software Development projects which represents the components of coordination effectiveness. Strode and Huff reveal that expertise is one of the coordination components in Agile Software Development.

Specific to expertise coordination, Maruping et al. [133] focus on how expertise coordination and Extreme Programming (XP) practices interact to affect performance in Agile Software Development projects. Maruping's study reveal that there was a significant interaction between XP practices and expertise coordination. The findings indicate that expertise coordination exists in Agile Software Development projects. Along with Strode's study, Maruping and her colleagues' study has also provided a motivation to explore expertise coordination in Agile Software Development projects in-depth.

An Agile team is a cross-functional team that includes all the expertise necessary for every phase involved in developing software [207]. In order to have all the expertise needed, it is not feasible for all individuals with relevant expertise to be part of a team [194]. In certain circumstances, Agile teams need to rely on other expertise which is located outside the team, such as user experience designers and database administrators [194]. This is a point where expertise coordination is vital to manage expertise dependency between Agile teams and roles outside Agile teams and where a further investigation is needed. Therefore, this study aims to explore how to manage and utilize internal and external expertise in Agile teams through expertise coordination.

Even though many studies emphasize coordination in Agile teams, there is a paucity of empirical studies that focus on expertise coordination. Abrahamsson et al.[1] suggest further research is needed to explore the human resource dependencies faced by Agile teams, including expertise. There is clearly a need to conceptualize and theorize the underpinnings of expertise coordination from an Agile Software Development perspective.

1.1 Research Objectives

The aim of this study is to build a theory of expertise coordination in Agile Software Development projects, which is guided by three research questions:

- **How is expertise coordinated in Agile Software Development projects?** This question attempts to understand the expertise coordination process that should be carried out by Agile team members in managing expertise dependencies.
- **What are the strategies used in coordinating expertise outside Agile teams?** This question attempts to identify strategies used in coordinating expertise outside Agile teams.
- **What are management strategies support expertise coordination in Agile Software Development projects?** This question attempts to determine the roles of management in supporting expertise coordination in Agile Software Development projects.

1.2 Structure of the Thesis

The remainder of this thesis consists of the following chapters:

- **Chapter 2 : Literature Review** – This chapter presents the theoretical underpinnings of the study, including expertise coordination in general, from the software development perspective, and in the context of Agile Software Development. This chapter also reviews studies of expertise coordination.
- **Chapter 3 : Research Methodology** – This chapter provides a justification of Grounded Theory as a research method for this study, and a description of how we carried out this study by employing

Grounded Theory. This chapter describes how the data was collected and analyzed using the Grounded Theory.

- **Chapter 4 : The Theory of Expertise Coordination** – This chapter presents the key contribution of this study: expertise coordination in Agile Software Development projects, introduces emergent categories and relationships, and presents the theory boundaries that are reflected by its emergent categories and relationships.

- **Chapter 5 : Expertise Coordination Processes** – This chapter presents the expertise coordination process in Agile Software Development projects.

- **Chapter 6 : Coordinating Expertise Outside Agile teams** – This chapter describes how Agile teams coordinate outside expertise through strategies to manage external expertise.

- **Chapter 7 : Management Support** – This chapter describes how management support influences expertise coordination in Agile Software Development projects.

- **Chapter 8 : Discussion** – This chapter reviews the research findings and then describes the relationships between categories that lead to the emergent theory. This chapter then provides a detailed explanation of how the results relate to both theoretical foundations and the existing literature, and describes how we evaluate and validate the emergent theory.

- **Chapter 9 : Conclusion** – This chapter summarizes the contributions of this study to theory and practice, discusses the limitations of this study and of the emergent theory, and outlines several suggestions for future work.

2

Literature Review

This chapter reviews the literature on expertise coordination in general, then continues with an explanation of expertise coordination from the software development perspective. Since this research is carried out in the context of Agile Software Development, this chapter also presents an overview of Agile Software Development. Finally, this chapter explains expertise coordination in Agile Software Development projects.

2.1 Expertise Coordination

Engaging in expertise coordination research requires a researcher to clearly understand the basis of expertise coordination. Before defining expertise coordination, two fundamental concepts of expertise and coordination are discussed in this section.

Expertise refers to skills and knowledge that are retained by individuals to produce an outstanding performance [147], plus the ability of a person to generate knowledge in solving specific problems [79, 104]. Based on

this definition of expertise, skills and knowledge are the main elements of expertise. A skill is an ability to perform a specific task, while knowledge consists of the information needed for the skill [27].

Experts and highly skilled staff possess expertise, and their expertise is not based on the amount of knowledge and skills accumulated in a specific domain, but rather on their interactions in utilizing that expertise [186]. Besides individual skills and knowledge, the construct of expertise consists of how experts interact with others in demonstrating and sharing their expertise. This is the point where coordination is essential to support varied skills and expertise levels, particularly at team levels.

There are a variety of definitions of 'coordination' terms proposed by scholars. Table 2.1 depicts selection of 'coordination' definitions from different research fields. Most 'coordination' definitions, however, indicate commonalities as follows:

- an ability to organize and structure transactions between interdependent components, such as tasks, people, and resources.
- an ability to working together with a common understanding to achieve shared goals.

Coordination can be seen through the dependencies between people who work together with a common understanding to build software efficiently [102]. Strode et al. [204] propose the existence of implicit coordination, which encompasses five components: *'Know why','Know what is going on and when', 'Know what to do and when', 'Know who is doing what'*, and *'Know who knows what'*. The last component, *'Know who knows what'* indicates expertise as a coordination component. Expertise is prevalent in team knowledge and, to be applied in projects, the expertise must be managed effectively through expertise coordination.

Expertise coordination is defined as "*the management of knowledge and skills dependencies*" [54]. This definition indicates how team members should depend on each other in managing and utilizing their expertise resources.

Table 2.1: Several Definitions of 'Coordination'.

Author	**Definition**
Faraj and Xioa [55]	Coordination is about the integration of organizational work under conditions of task interdependence and uncertainty
Quinn and Dutton [170]	Coordination is the process through which people arrange actions in ways that they believe will enable them to accomplish their goals
Argote [11]	Coordination involves fitting together the activities of organization members, and the need for it arises from the interdependent nature of the activities that organization members perform
Kraut and Streeter [102]	Different people working on a common project agree to a common definition of what they are building, share information, and mesh their activities
Van de Ven et al. [216]	The integration or linking together of different parts of an organization to accomplish a collective set of tasks
Kotlarsky et al. [101]	Coordination is then perceived as a problem of sharing, integrating, creating, transforming, and transferring knowledge
Malone [130]	When multiple actors pursue goals together, they have to do things to organize themselves that a single actor pursuing the same goals would not have to do. We call these extra organizing activities coordination
Malone and Crowston [129]	Coordination is the managing of dependencies between activities

Expertise coordination requires the team to recognize who has particular expertise, when and where they are needed, and how to access the expertise effectively [54]. Therefore, it is important to identify who has particular expertise and recognize the need for the expertise. The identified expertise then can be identified and accessed when needed in a timely manner. At this point, appropriate mechanisms are vital to enable the expertise to be accessed effectively.

In order to utilize expertise, Garrett and his colleagues [61] have presented a set of descriptions of expertise, which consist of the following dimensions:

- Expertise - understanding knowledge in a specific area by answering 'what' and 'how' something works in a specific knowledge domain;
- Situational context expertise - recognizing environmental and situational demands by answering 'when', 'where', and 'why' particular knowledge and skills are relevant;
- Interface tool expertise - utilising a specific interface tool to apply a particular subject-matter expertise;
- Expert identification expertise - knowing who has a particular expertise area and level;
- Communication expertise - transmitting knowledge effectively via appropriate media;
- Information flow path expertise - knowing when a communication path is available within a specific task and situational context.

These six dimensions are used to coordinate expertise at both individual and/or team levels [61]. Every individual should clearly define their subject matter expertise and situational context expertise, in order to know what their skills are and when and where the skills can be applied in accomplishing tasks. As social connection is an integral part of a team,

every individual should be aware of other expertise areas and levels. This enables the process of transmitting expertise to others by knowing the source of available and relevant expertise.

Strode [205] summarizes coordination in several categories based on organisation theory, coordination theory, information projects, teamwork, and Agile Software Development. Table 2.2 depicts coordination categories which are adapted from Strode [205].

Strode [205] categorizes expertise coordination under cognitive coordination. Cognitive coordination occurs when team members perform tasks and need to understand other team members, and to adjust their behaviour in order to cope with others [176]. Teamwork and collaboration are basic requirements that lead to cognitive coordination. Cognitive coordination is often known as implicit coordination, since the coordination relies on tacit knowledge without emphasizing explicit knowledge. Shared mental models [117], collective mind [45], and expertise coordination [54] are classified under cognitive coordination as they are forms of implicit coordination.

Other coordination categories, however, indirectly exhibit expertise coordination such as relational coordination. Relational coordination involves human interrelationships and is defined as *"frequent, timely, problem-solving communication"* [174]. Expertise coordination involves interdependencies between team members in managing expertise resources and this can be categorized under relational coordination.

Expertise coordination has been studied in a variety of research domains such as in software development [54, 133, 195], banking [100], flight reservations [10], event response [28], emergency medical response units [55], and emergent groups responding to disasters [128]. Each study emphasizes different aspects of expertise coordination, and the research methods vary depending upon the research objectives.

Kotlarsky et al. [100] investigated the role of 'codification' in coordinating expertise between client, staff, and onsite and offshore vendor personnel in a large-scale outsourcing contract. Codification is important to

Table 2.2: Categories of Coordination Adapted from Strode [205].

Coordination Category	**Description**
Structural coordination	Formal coordination structures (hierarchies, roles, and responsibilities) [101, 143, 216] Example: impersonal mode (rules and procedures) and mutual adjustment (group or personal meetings)
Technological coordination	Coordination achieved by using software tools [16] Example: social network applications, and email
Spatial coordination	Coordination achieved through team settings [127, 194] Example: co-located teams and open workspace
Cognitive coordination	Coordination achieved when team members anticipate actions, needs, and understand others implicitly [45, 54, 117, 176, 228] Example: shared mental models, a collective mind, and expertise coordination
Relational coordination	Coordination achieved by human interrelationships and focuses on problem solving [65, 174] Example: Face-to-face communication, regular meetings
Coordination by artefacts	Coordination achieved through existing artefacts' solving [129, 194, 174] Example: whiteboards, and index cards
Coordination by activities	Coordination achieved through Agile practices [148, 194] Example: stand-up meeting, refactoring, and collective ownership

support expertise coordination, which enables the process of externalizing expertise in explicit form, as well as storing and transferring expertise [231]. The contribution of Kotlarsky et al.'s study is the extension of 'codification' theory, which explicitly demonstrates the role of 'codification' in supporting expertise coordination through two codification practices: knowledge codification, and codification of the knower.

Expertise coordination involves socially shared cognitive processes to enable the knowledge and skills dependencies [54]. Transactive Memory Systems (TMS) provide a dynamic of human connection which fosters socially shared cognitive processes [6]. TMS exists when two or more people interact cooperatively in storing, retrieving, and communicating information [226]. Wegner [226] conceived the term of TMS, which focuses on integrating and utilizing expertise in order to optimize the value of members' knowledge [118].

In order to improve disaster response, Majchrzak et al. [128] used Transactive Memory Systems (TMS) theories to coordinate knowledge, people, resources, tasks, and technology efficiently. TMS is a theory of knowledge coordination which describes interconnected individual memory systems in sharing knowledge [227]. TMS occurs through shared mental models, which consists of a directory of who knows what and a process of encoding and retrieving information.

As the effects of TMS on emergent response groups were different from stable groups, Majchzak et al. [128] found the need to extend TMS to deal with coordinating knowledge without a shared mental model, since emergent response groups posed challenges in managing the roles involved. The shared mental model was replaced with community developed narratives to describe the events or scenarios involved in knowledge coordination.

On the other hand, Akgun and his colleagues [5] determined the antecedents and consequences of TMS in new product development projects. TMS assists new product development projects which indirectly tends to support expertise coordination as follows:

- Reducing the cognitive load in remembering each member's expertise.
- Accessing a pool of expertise.
- Facilitating the process of sharing tacit knowledge of different domains.
- Allocating resources according to member expertise.

The findings of Akgun et al.'s study [5] reveal that TMS relies on team stability, team members' familiarity, and interpersonal trust. This study also indicates the positive impact of TMS on team learning, speed-to-market, and success of new product development.

Fast-response organizations face coordination challenges to operate under high uncertainty conditions and fast decision making [55]. Faraj and Xioa [55] used these challenges as a motivation to investigate coordination practices of medical trauma centres. The expertise coordination practices represent practices used to manage knowledge and skills dependencies as follows:

- Reliance on protocol - to structure interactions and manage resources and expertise dependencies.
- Plug and play teaming - to manage patient knowledge and expertise interdependencies when new patients are brought in.
- Communities of practice - to manage staffing interdependencies and the learning process.
- Knowledge sharing - to ensure knowledge is shared without error.

The research on expertise coordination is summarized in Table 2.3. The next subsection discusses expertise coordination in the software development context.

Table 2.3: Examples of Expertise Coordination Research

Researcher	Research Domain	Research Objectives	Research Findings
Kotlarsky et al. [100]	Banking	To examine the role of 'codification' in coordinating expertise between client, staff, and onsite and offshore specialists.	Reconceptualised the 'codification' term.
Majchrzak et al. [128]	Emergent groups responding to disasters Domain	To extend the roles of TMS	Proposed TMS extension for coordinating expertise in emergent response groups
Faraj and Xioa [55]	Fast-response organizations Domain	To develop expertise coordination practices	Proposed 4 practices in managing expertise dependencies: reliance on protocol, plug and play teaming, communities of practice, and knowledge sharing
Akgun et al.'s [5]	New product developments Domain	To determine the antecedents and consequences on the TMS	Revealed antecedents of TMS (team stability, team member familiarity, and interpersonal trust) and consequences of TMS (team learning, speed-to-market, and new product success).

2.2 Expertise Coordination in Software Development

Software development projects rely on expertise in producing quality software products [180, 229]. Expertise is a critical resource and it is important to ensure the presence of sufficient knowledge and skills for software development teams [54]. In producing a quality software product, the presence of expertise is not adequate on its own [54]; rather it is important to leverage the available expertise through expertise coordination.

Faraj and Sproull [54] embarked on an expertise coordination study specifically in the software development context. Faraj and Sproull affirm the importance of expertise coordination in leveraging potential expertise [54]. Their findings reveal that there is a strong relationship between expertise coordination and team performance. Team performance does not just depend on the available expertise, but also on how team members depend on each other in utilizing that expertise effectively.

Team members divide the cognitive labour for their tasks with individual specialization in different domains [226]. Team members are responsible for possessing relevant expertise for their tasks. This is a point where TMS are needed to enable team members to rely on each other in sharing and utilizing their expertise resources [226]. The successful TMS rely on the ability of team members to identify and recognize the knowledge possessed by other and access the particular expertise.

Based on TMS, Faraj and Sproull [54] assert three processes of expertise coordination: locating expertise, recognizing expertise, and retrieving expertise.

Locating expertise requires knowing the area of expertise of other team members. Knowing "*who knows what*" leads to identifying who has particular expertise [54]. Team members should have meta-knowledge of the available expertise in their team. Every team needs to develop a common language for describing tasks and roles that lead to the location of expertise.

This is supported by Sandra and her colleagues [62] who claim that every team member should have awareness of others' expertise, particularly the relevant expertise to perform tasks. Knowing where to get an answer for solving a problem assists team members in determining their peers' expertise [54].

After locating expertise, it is important to recognize the necessity of the identified expertise [54]. Recognizing expertise involves a process to identify where and when the available expertise is needed [54]. There is no point having massive expertise available if team members fail to recognize the condition for the necessity of the expertise.

Ultimately, expertise coordination enables team members to retrieve existing expertise to solve ad-hoc problems. Each team member should know an effective way to access relevant expertise in a timely manner [54]. Integrating informal interaction and problem solving tends to bring the right expertise when it is needed. Instead of verbal communication, informal interaction also involves nonverbal communication such as eye contact, body gestures, and facial expressions to strengthen the expertise retrieving process [87].

These processes represent general activities involved in managing expertise resources effectively, which are not to be executed in a rigid sequence. Faraj and Sproull [54] propose three criteria of expertise coordination processes:

- distributed - to manage dispersed expertise among team members
- heedful - to manage overlapping tasks knowledge among team members
- emergent - to manage unspecified solutions for ad-hoc problems through effective interactions.

Lewis [118] affirmes that the expertise coordination study by Faraj and Sproull [54] is similar to TMS, particularly in locating expertise. Faraj and

Sproull's study emphasizes locating expertise rather than how to coordinate the expertise. There is no discussion on the mode or medium to coordinate the expertise. They suggest that further investigation is needed to explore expertise coordination processes.

Specifically in software development projects, Faraj and Sproull affirm that expertise coordination has a positive impact on team performance. Referring to Faraj and Sproull's study, Shim et al. [195] conducted research on expertise coordination by examining expertise coordination mediating between user-information system coproduction and development project outcomes. Coproduction involves user participation in the software system development since their domain knowledge is essential for effective software development. Coproduction requires integration of users' and developers' expertise in producing desired software project outcomes. Besides providing software system requirements, users also have responsibility to collaborate with developers in developing the software. Hence, coproduction requires effective expertise coordination to locate and apply expertise through the software project development.

2.3 Agile Software Development

In the early 1960s, the software engineering community faced a software development crisis when focusing on large and long-lived software projects [197]. In order to produce better software, project planning, formalised quality assurance, and controlled and rigorous software development processes were thought essential through 'heavyweight' methods [2, 13, 155, 197].

Dissatisfaction with heavyweight approaches led to the evolution of 'lightweight' software development approaches known as Agile Software Development. In February 2000, Kent Beck and several software leaders proposed Agile Software Development with the following 'lightweight' characteristics [59]:

> *"We want to restore a balance. We embrace modelling, but not in*

> *order to file some diagram in a dusty corporate repository. We embrace documentation, but not hundreds of pages of never-maintained and rarely-used tomes. We plan, but recognize the limits of planning in a turbulent environment".*

Agile Software Development is a group of software development methods that promote iterative and incremental development [108]. The Agile Manifesto defines Agile Software Development with the following four core values [59]:

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

The first core value states that people are the most important factor rather than processes and tools. Agile Software Development emphasizes people by concentrating on competency development [39]. This value encourages Agile team members to interact and collaborate together in developing software [39]. Many studies have carried out human and social aspects specifically in the Agile Software Development context [83, 179, 194, 205, 232].

The second core value indicates that delivering a working software product is a priority in Agile Software Development teams, rather than presenting a pile of documents to customers [59]. Agile Software Development emphasizes the actual software implementation, and produces documents that only neccessary to serve the design and the deployment of the working product.

The third core value emphasizes customer involvement. In the Agile approach, on-going and regular meetings with customers are considered essential to capture requirements. A contract is useful for determining the

condition and boundaries of Agile Software Development [59], but relying on a contract does not guarantee the delivery of software that fulfils the customer's needs without regular customer involvement.

The last core value affirms that Agile teams need to adapt and respond to changing requests. Customers have a right to give feedback to improve a software product.

Behind the Agile manifesto there are 12 principles that underpin the common practices in Agile Software Development [59]. The principles act as guidelines for implementing Agile Software Development projects. These Agile principles are as follows:

- *Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.*
- *Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.*
- *Deliver working software frequently, from a couple of weeks to a couple of months, with a preference for the shorter timescale.*
- *Business people and developers must work together daily throughout the project.*
- *Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.*
- *The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.*
- *Working software is the primary measure of progress.*
- *Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.*
- *Continuous attention to technical excellence and good design enhances agility.*

- *Simplicity is essential – the art of maximizing the amount of work not done.*
- *The best architectures, requirements, and designs emerge from self-organizing teams.*
- *At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behaviour accordingly.*

There are a variety of Agile methods such as Extreme Programming (XP) [19], Scrum [187, 207], Lean Development [168], Adaptive Software Development (ASD) [82], Dynamic Systems Development Methodologies (DSDM) [199], FDD (Feature Driven Development)[163], and Crystal family [38]. Scrum and XP are considered to be the most widely adopted Agile methods [219] and are discussed in detail in the next subsections.

2.3.1 Scrum

Scrum is an iterative and incremental Agile method for managing software development projects [187]. This Agile method was developed by Sutherland in 1993 and two years later officially formalized by Schwaber [207]. The name 'Scrum' was inherited from the groundbreaking paper '*The New Product Development Game*' by Takeuchi and Nonaka in 1986 [210].

Scrum implements iterations called sprints to structure the development cycles [187]. Each iteration can be up to four weeks. Sprints are time-boxed, which means the team has a fixed start and end date to make a commitment for completing the sprint goal. If the team is unable to accomplish the goal, there is no extension allowed. Furthermore, additional functionalities cannot be added during sprints.

Each sprint requires three primary roles [187]:

- Product owner – A product owner is a customer representative. The product owner needs to identify the product backlog that consists of software features. The product backlog needs to be prioritized by the product owner.

- Scrum team – A team consists of between six and eight people. Every team needs to do all relevant tasks including planning, analysis, programming, and testing the software product.

- Scrum master – A Scrum master serves the team as a mentor to guide, facilitate, and educate the team members in applying Scrum practices. The Scrum master is responsible for coordinating and conducting Scrum meetings such as sprint planning meetings, daily stand-up meetings, sprint reviews, and sprint retrospectives.

During sprint planning, the Scrum team and the product owner review the product backlog facilitated by the Scrum master. The Scrum team selects work items from the prioritized product backlog. The selected items form sprint backlogs.

Once the sprint has started, Scrum team members attend daily stand-up meetings. A stand-up meeting is a daily meeting and is conducted within 15 minutes or less. The purposes of daily stand-up meetings are to coordinate tasks among team members and provide a medium for communication [187, 115]. This meeting gives opportunities for Scrum team members to report three important issues as follows [187, 207]:

- What they did since the last meeting.

- What they plan to do by the next meeting.

- What obstacles they had faced in finishing their tasks.

At the end of each sprint, there are two main 'ceremonies': sprint review and sprint retrospective [207]. A sprint review provides a platform to inspect what has been done during the sprint. During the sprint review, the team needs to demonstrate the product increment to the following parties [187]:

- Management - to see the progress of the team with the resources provided.

- Customers - to see what the team has built according to their needs
- Product owner - to see how much functionality has been built

A sprint review meeting might be attended by other team members or parties to see the product increment from many different perspectives [187]. For instance, developers tend to see what the team was able to do with the available technology. Based on the product demonstration, they can gain a similar understanding on the status of the product backlogs, which indirectly leads to identifying drawbacks that need to be fixed. They can then make decisions on how to fix the drawbacks and what to do next.

While a sprint review focuses on the product, a sprint retrospective reviews the process of development. The sprint retrospective is used to identify tasks or activities that did not run smoothly and discuss how to improve performance. Instead of running one retrospective at the end of a project, Scrum teams are able to reflect upon the current software processes immediately after each sprint review and define recommendations for improvement [103].

2.3.2 Extreme Programming (XP)

XP was introduced by Kent Beck [19] in 2001 and emphasizes short iterations. The word "*Programming*" in its name indicates that XP focuses on programming tasks rather than on management practices [115]. XP introduces strict coding practices to ensure the development of a quality software product [115].

XP originally had four values; one more was added later [20]. The five values are as follows [20]:

- Communication – XP promotes continual communication among all team members including customers, managers, and development teams. Maximizing communication is important to disseminate knowledge and form a mutual understanding among team members.

- Simplicity – XP encourages starting with simple design and coding, and provides space for adding additional features. Simplicity values avoiding complicated solutions and embellishing when needed.
- Feedback – XP encourages Agile team members to get feedback from the customer, peers, and the system. Early, frequent, and clear feedback is vital to improve the software development product.
- Courage – From an XP perspective, courage refers to persistence in addressing any issues that arise or affect the selected task.
- Respect – XP encourages the Agile team members to respect each other as well as respecting themselves. Self-respect requires striving to develop quality work.

XP promotes 12 practices in developing software as follows [19, 122]:

- Planning game – The planning game is a planning process in XP, which determines the scope of each iteration by defining the requirements, activities, and tasks. User requirements are presented as user stories in index cards. The cards are placed on a wall to display task visibility and reflect progress.
- Pair-programming – Pair-programming involves two team members working together on the same task from design to testing.
- Small release – XP focuses on delivering incremental functionality based iterations.
- Collective ownership – Every team member has the right and responsibility to modify the code at any time, and add software functionality as well. Team members are also responsible for committing all changes to the code repository.
- Simple design – XP promotes simple design and solutions rather than complex solutions.

- Refactoring – In order to improve the software, programmers can modify the code without changing the code's behaviour.

- Continuous integration – XP encourages programmers to integrate the software quickly to avoid conflicts later.

- Metaphor – Metaphor assists Agile team members and customers to understand the whole picture of software development.

- Testing – Programmers do their own unit testing and customers need to participate in acceptance testing. Testing is a way to gain feedback in improving the software deliverables.

- Coding standards – Coding standards provide consistency to improve communication among team members.

- 40-hour week – In order to maintain productivity, no one in an Agile team can work more than 40 hours per week. Agile team members can work overtime only when it is effective.

- On-site customer – An XP team consists of five to ten programmers and a customer representative. Customers are integral parts of XP teams and must be readily accessible to answer any questions that arise from programmers.

Since Scrum focuses on management practices, some Agile practitioners adopt a hybrid Agile method by integrating XP and Scrum [151]. This is clearly stated in the "*7th Annual state of Agile development*" [219], in which Scrum and XP hybrid was found to be the second most popular Agile method.

2.3.3 Expertise Coordination through Agile Practices and Artefacts

According to Strode [205], expertise coordination can be categorized under coordination by artefacts and coordination by activities (refer to Table 2.2). Several studies have shown that Agile practices and Agile artefacts have positive impacts on coordination [133, 194, 205]. As expertise is one of the coordination components, the importance of expertise coordination also can be seen through Agile values and practices. Table 2.4 summarizes how expertise coordination would appear to be relevant in Agile Software Development.

Face-to-face communication is the most efficient and effective way to convey information among Agile team members [59]. Agile teams predominantly rely on tacit knowledge through face-to-face communication [179, 194]. Tacit knowledge is knowledge that is very hard to codify or extract compared to explicit knowledge [57, 63]. As knowledge is a part of expertise, expertise coordination is a prevalent mechanism to manage tacit knowledge dependencies, particularly to retrieve the tacit knowledge effectively.

Agile Software Development focuses on producing a working software product rather than comprehensive documentation [20]. As a result, less documentation is produced since more time is spent on delivering working software, which impacts on organizational memory in Agile teams. In order to support the organizational memory, Agile methods shift the organizational memory from external to tacit knowledge by replacing documents with informal communication [37]. This is a point where expertise coordination is vital to strengthen communication in supporting the organizational memory in Agile teams.

Agile Software Development emphasizes active customer involvement, particularly in providing and prioritizing user requirements, and in refining product features [72, 131]. Collaboration between customers and the soft-

Table 2.4: Expertise Coordination through Agile Values and Practices.

Agile values/ practices	**Expertise coordination**
emphasizing face-to-face communication [59].	to manage tacit knowledge dependencies [179, 194].
producing a working software product rather than comprehensive documentation [20]	to support the organizational memory [37]
emphasizing active customers' involvement [72, 131]	to transfer software domain knowledge and customers' expertise into software development teams [33, 195]
coping with unpredictable changes of requirements [19]	to identify the right expertise to deal with software change requests [19]
cross-functional teams [207]	to manage expertise outside Agile teams and assist the growth of a *"T-shaped person"* [83, 88, 194]
self-organizing teams [81]	to identify and quantify the level of team members' expertise in selecting tasks [83]

ware development team is vital to disseminate software domain knowledge [33, 195]. In order to capture user requirements, an effective mechanism is needed to enable customers to transfer their domain knowledge and merge with the expertise of the software development team. Shim et al. [195] affirm the need for expertise coordination in strengthening knowledge transfer between customers and software development teams.

Agile Software Development encourages the delegation of authority to Agile teams in dealing with requests for changes [19]. The delegation of authority includes decisions in identifying the right person to make software changes based on the customers' requests [19]. Identifying the

right expertise through expertise coordination helps teams respond to changing user requirements effectively.

Agile Software Development promotes cross-functional teams which include all the expertise necessary for every phase of developing software [207]. In certain circumstances, Agile teams need to rely on other expertise which is located outside the team such as user experience designers and database administrators [194]. Expertise coordination is vital to manage expertise dependency between Agile teams and roles outside the teams.

A cross-functional team needs all team members to understand each others' role and participate beyond their area of expertise [83]. They do not just rely on their expertise in choosing tasks, but also tend to perform other tasks when needed. This requires Agile team members to learn about others' expertise. Each Agile team member needs to be a multi-skilled person, which is commonly known as a "*T-shaped person*" [24, 95]. A "*T-shaped person*" possesses a deep vertical skill in a specialized area, as well broad skills in other areas of expertise. Figure 2.1 illustrates a group of *T-shaped people* in an Agile team, where each member complements one another by broadly knowing others' skills. The effective way to become a "*T-shaped people*" is by engaging in a new task, which requires new knowledge and skills. Collaborating with peers could lead Agile members to learn quickly from other team members [88].

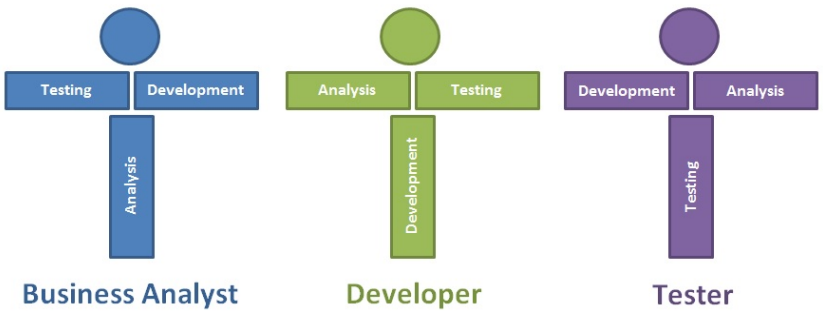


Figure 2.1: *"T-Shaped Person"* in an Agile Team Adapted from Clokie [36].

There are several reasons for building an Agile team with "*T-shaped person*". The main reason is to ensure Agile teams mutually adapt to the immediate needs of the software development [9]. For example, if a software tester has a large workload, he or she could be assisted by an experienced business analyst to prevent a bottleneck. At the same time, if a software developer faces a problem in debugging, he or she could ask for assistance from other team members. Expertise coordination is needed since Agile team members depend on other team members' expertise in accomplishing the team's tasks.

Besides cross-functional teams, Agile teams are expected to be self-organizing. A self-organizing team provides flexibility and authority for Agile team members to organize their own work [81], and practices self-assignment in delegating tasks. This means that every team member has freedom in selecting their tasks. Selecting tasks is based on the task's priority estimation and individuals' capabilities [81]. Hoda et al. [83] reveal that some team members choose a task by considering others' capabilities and avoiding the selection of tasks based on ease of implementation. The finding indicates that expertise coordination is needed for a self-organizing team to identify individual capabilities and levels of expertise.

2.4 Expertise Coordination in Agile Software Development

A number of researchers have studied coordination in Agile Software Development. Research has recognized coordination as a crucial aspect of Agile Software Development [84, 86, 144, 149, 169, 194]. For instance, Moe et al. [149] concede that coordination is a critical success factor for achieving Agile team effectiveness.

Through a Grounded Theory study, Hoda et al. [84] identify the role of coordinator in Agile teams. A coordinator is responsible for coordinating

the interaction and collaboration with teams and customer, particularly when dealing with software change requests. This finding clearly affirmes coordination as a crucial aspect of Agile Software Development which leads to the need of a 'coordinator' role.

Pries-Heje and Pries-Heje [169] posit coordination as one of the Scrum mechanisms that lead to successful Scrum. Through a longitudinal case study of Agile software projects based in Denmark and India, they found that Scrum provides a framework that supports coordination in Agile teams through Product Backlogs and Sprint Backlogs.

Cao and Ramesh [29] used the existing theory of coordination to examine the consistency between coordination mechanisms of Agile methods and coordination mechanisms proposed by Van de Ven et al. [216]. Their findings reveal there is a consistency between coordination modes from organization theory and Agile practices in small projects. As Agile methods emphasize small and medium-sized projects, Coa and Ramesh [29] propose implications for coordination practices that are different from traditional development.

Coordination is more challenging in distributed Agile Software Development than in co-located development [86]. Distance is a fundamental barrier in coordinating distributed software development [78]. Hole [86] carried out a study to understand how distributed Agile Software Development projects coordinate their work. Hole proposes several approaches to distributed Scrum, and also indicates that geographical transparency is essential for supporting mutual adjustment.

In the context of Agile Global Software Development, Li and Maedche [119] conducted a case study to examine which situational factors influence the formation of effective coordination strategies. Based on the coordination theory proposed by several researchers including Faraj and Sproull [55], they developed a deeper understanding of effective coordination strategies in Agile Global Software Development. Their main findings indicate that there is a consistency between situational factors in conventional and Agile

Global Software Development.

Besides team and project levels, coordination should be taken into account at the inter-team level. Paasivaara and her colleagues [162] focused on how to achieve inter-team coordination in large-scale Agile projects. They propose additional principles which to implement Scrum-of-Scrum meetings in order to ensure successful inter-team coordination.

Effective coordination, communication and collaboration often gain attention in Agile research [144, 167, 194]. Mishra and Mishra [144] investigated how the physical environment and Agile artefacts such as half-height cubicles and status boards play an important role in supporting coordination, communication, and collaboration. They report that half-height cubicles allow Agile team members to be aware of the status of inter-related tasks in teams with information provided by a status board. The status board enables Agile teams to clearly define tasks at the team and project level, and coordinate their tasks.

For the same purpose, Sharp and Robinson selected story cards and walls to investigate how these Agile artefacts facilitate the team's collaboration, communication, and coordination activities. Besides Agile artefacts, Sharp and Robinson [194] claim that Agile practices such as planning games, daily stand-up meetings, and pair-programming also have a positive impact on coordination.

Pikkarean and her colleagues [167] applied coordination theory by Malone and Crowston [129] to analyse the impacts of XP and Scrum practices used in communication. Their study indicates that Agile practices used in software development teams have a positive impact on communication. One surprising point arising from this study is that XP practices such as pair-programming and continuous integration do not affect team coordination.

Most studies emphasize determining the Agile practices and artefacts that have a significant influence in achieving coordination. A few empirical studies, however, focus on developing a coordination theory. Strode [205] proposes a theoretical model of coordination in the Agile Software Develop-

ment context. The theoretical model includes a coordination effectiveness part, which represents the components of coordination effectiveness. There are several coordination effectiveness components such as "*know who knows what*" that will determine who has which sort of expertise. Besides tasks and processes, Strode et al. [205] indicate that expertise is one of the which coordination components in Agile Software Development.

Yu and Petter [233] affirm the functionality of Agile practices in coordinating expertise. Yu and Petter [233] used a shared mental model to examine how Scrum and XP practices (system metaphor, stand-up meeting, and on-site customer) improve collaboration within software development teams. A shared mental model is related to a team's interaction and coordination including roles, knowledge and skills interdependencies. Their findings indicate that effective use of Agile practices tend to improve the quality of communication and coordination, which enable Agile team members to have a clear picture of each member's roles and expertise.

Specific to expertise coordination, Maruping et al. [133] focus on how expertise coordination and Extreme Programming (XP) practices interact to affect performance in Agile Software Development projects. Collective ownership and coding standards were selected to examine how XP enables Agile teams to coordinate expertise [133]. Collective ownership requires every team member to take ownership of the whole code by understanding others' work, whereas coding standards lead to the development of similar mental structures on coding processes. Both practices facilitate collaboration and knowledge sharing between team members, which tends to influence the process of locating, recognizing, and accessing the right expertise. Maruping et al.'s study reveal that there is a significant interaction between XP practices and expertise coordination. These findings indicate that expertise coordination exists in Agile Software Development projects and this has provided motivation to explore this research area in depth.

3

Research Methodology

This chapter describes the research methodology that we used to carry out this study. This study applied a qualitative research method to deal with a non-technical aspect of software engineering. Therefore, this chapter begins with an overview of qualitative research, then continues with a general explanation of common qualitative research paradigms: positivist, interpretivism, and critical theory. A general explanation of qualitative research methods follows. Grounded Theory was selected as a research method for this study, and the reasons underpinning the choice of Grounded Theory will be discussed next. Finally, this chapter thoroughly explains the Grounded Theory procedures that were employed in this study.

3.1 Qualitative Research

Qualitative research is an exploration in understanding social phenomena in depth [44, 48, 75, 166]. Qualitative research involves interpretation of human feelings, opinions, and experiences [44, 48, 75, 166]. Qualitative

researchers need to describe the research as it occurs in the research context without attempting to control or manipulate the subject matter [75]. In software engineering, qualitative research has been accepted, particularly in dealing with human factors research [111, 189]. This study focuses on understanding human thoughts, actions, and social interactions between Agile team members in coordinating their expertise, and expertise outside Agile teams as well; therefore, the qualitative approach is applicable for this study.

3.2 Qualitative Research Paradigms

Engaging in qualitative research requires researchers to clearly understand their research paradigms. Guba and Lincoln [74] define research paradigms as *"basic belief system or worldview that guides the investigator"*. A research paradigm represents a view for researchers to understand the nature of the research, a relationship between researchers and participants, and methods to extract and understand knowledge that is related to the research [74]. Research paradigms can be described through ontological, epistemological, and methodological 'lenses' as below [44, 74]:

- **Ontological** – The participants' view of the nature of reality.
- **Epistemological** – The relationship between the researcher and research subjects.
- **Methodological** – The process of research.

Three common research paradigms are considered in this study:

Positivism : A positivist view assumes that reality is factual and objective [161]. In order to understand phenomena, reality can be described by using independent measurable properties [74, 152]. A positivist researcher should not intervene in the phenomenon of interest [74]. The positivist

researcher attempts to test theory by verifying hypotheses [74]. A case study is an example of a research method within a positivist paradigm.

Critical Theory : A critical theory paradigm attempts to critique an existing social system or status quo [161]. Analyzing the participants' views critically is important for the critical theory researcher, rather than simply accepting the participants' view. A critical theory researcher tends to choose a long-term study such as ethnography.

Interpretivism : A interpretivist view assumes that reality is a social world [161]. The social world is reinforced through human action and interaction. Interpretivist research attempts to understand individuals' meanings such as perceptions, feelings, and opinions within their social settings. A interpretivist researcher avoids imposing any predefined dependent and independent variables, and the researcher attempts to derive categories and themes that are closely relevant to the participants. Grounded Theory is an example of a research method within a interpretivist paradigm.

This study is based on the interpretivist paradigm for the following reasons:

- This study attempts to deal with human interaction in the Agile Software Development context. The aim of this study is to investigate how Agile team members depend on each other in utilizing their expertise.
- This study focuses on developing a theory, rather than verifying an existing theory.

3.3 Qualitative Research Methods

There are several qualitative research methods such as phenomenology, ethnography, case study, and Grounded Theory:

Phenomenology: Phenomenology focuses on the study of phenomena including events, situation, experiences, and concepts [44, 75]. Phenomenology requires the researcher to describe, explicate, and interpret the essence of phenomena [44, 166]. For instance, a phenomenological study could describe the experiences of consumers in using a new product. The researcher needs to collect data from persons who are experienced in using the product. The primary data collection for phenomenology is through interviews with the participants [44, 166]. The researcher is also able to use other techniques such as observations and documents for data collection [44].

Ethnography: Ethnography is appropriate when describing and interpreting a group that shares the same culture [44, 75]. Ethnography emphasizes the naturalistic, which involves observation of individuals or group behaviour in a natural context. An example of an ethnographic study is the exploration of the values, beliefs, and assumptions that shape and sustain Agile practices [193]. In order to carry out this type of study, the researcher acts as an observer by engaging in day-to-day activities such as attending Agile meetings, pair-programming, and having lunch together. This research method is time-consuming because sometimes the researcher tends to spend long periods of time immersing themselves in the research field [75].

Case Study: A case study is appropriate for exploring issues or problems in depth through one or more cases within a context [44, 75]. This research method focuses on the event, activity, or programme rather than an individual [44, 75]. An example of case study is the exploration of impacts of pair-programming on the software product quality. In order to carry out this study, a researcher may use more than one software company as cases. In conducting the case study research, multiple data collection methods can be applied including observations, interviews, documents, and artifacts.

Grounded Theory: Grounded Theory consists of systematic guidelines for developing a theory 'grounded' in data [71]. This research method focuses on a process, action, or interaction among individuals [44]. The primary data collection technique is an interview with research participants [23, 44]. The researcher is also able to use other techniques such as observations, fieldnotes, and document analysis, depending on the topics and research objectives.

Grounded Theory is applicable as a research method for this study due to the following reasons:

- Grounded Theory has commanded attention and is widely used as a research method for developing a theory in many fields of study including software engineering [3, 51, 85, 224].

- Grounded Theory is appropriate for exploring human behaviour and social interactions [67]. This study focuses on how Agile team members rely on each other in coordinating their expertise including outside expertise, which involves human behaviours and interactions.

- Grounded Theory is appropriate to be used in areas that are underexplored which require further investigation [23]. Further investigation is needed to conceptualize and theorize about the underpinnings of expertise coordination on Agile Software Development perspectives.

- Grounded Theory is applicable for a study that emphasizes processes. Glaser and Strauss [71] argue that a basic social process is central and fundamental to Grounded Theory. The aim of this study is to explore the expertise coordination process for Agile Software Development projects, which is aligned with the characteristics of Grounded Theory.

3.4 Grounded Theory

Glaser and Strauss define Grounded Theory as *"the discovery of theory from data systematically obtained from social research"* [71]. Grounded Theory is an inductive research method that aims to infer new theories from observed data. A theory consists of general explanation of a process, an action, or an interaction shaped by the views of participants [66, 203]. As theory-building, Grounded Theory facilitates a substantive theory to emerge naturally from 'grounded' data within a specific area of enquiry.

There are several distinctive characteristics of Grounded Theory as follows:

- The main purpose of Grounded Theory is to develop a theory of processes or actions [66, 203].
- The theory should emerge naturally from data without forcing the data to fit preconceived ideas [71]. Researchers need to maintain an open mind without any pre-existing theoretical expectation [67].
- Grounded Theory invokes iterative strategies of going back and forth between data collection and analysis by using a constant comparison method [71].
- Theoretical sampling is important to determine the future data based on the current data analysis [71].
- Memoing is an integral part of Grounded Theory to formulate the flow and process that are being seen by researchers [66].

Glaser and Corbin [71] developed Grounded theory in the 1960s while studying death and dying in hospital. Based on observations of how dying occurred in a different hospital settings, they produced theoretical analyses by using a systematic research method. Through the research method, they advocated developing a theory from 'grounded' qualitative data rather

than testing hypotheses from existing theories. They finally entitled their innovative research method Grounded Theory.

Strauss and Corbin, however, proposed another variation of Grounded Theory in 1990. Consequently, this led to the emergence of two variations of the Grounded Theory method: traditional Glaserian and evolved Straussian [23]. There are many disagreements between Glaserian and Straussian, as depicted in Table 3.1 [160, 202].

These divergences mean that researchers need to be aware of which variation of Grounded Theory they use in their research [41, 214]. This study mainly employed the Glaserian approach due to a simple and uncomplicated coding process. The coding process is the most crucial part in Grounded Theory. The good coding process should lead to the emergence of reliable theory.

Strauss and Corbin [203] propose a different variation of Grounded Theory by considering the application of Grounded Theory in an educational setting. In order to fulfill academic requirements, this study also employed a few suggestions from the Straussian approach: reviewing the literature at the early stage of a Grounded Theory study and formulating research questions.

As depicted in Figure 3.1, the following steps outline the general process of Grounded Theory for this study adapted from Hoda et al. [85]:

- Defining a research area
- Continuous data collection
- Continuous data analysis
- Evaluating the theory

3.4.1 Defining a Research Area

The Glaserian approach requires the researcher to maintain an open mind without any pre-existing theoretical expectations [67]. This is to ensure

Table 3.1: Major Differences between Glaserian and Straussian Approaches as Adapted from Onion [160] and Stol et al. [202].

Glaserian	**Straussian**
Beginning with an empty mind	Having a general idea of where to begin
Emerging theory, with neutral questions	Forcing the theory, with structured questions
Development of a conceptual theory	Conceptual description
Theoretical sensitivity comes from immersion in the data	Theoretical sensitivity comes from methods and tools
The theory is grounded in the data	The theory is interpreted by an observer
Basic Social Processes should be identified	Basic Social Processes need not be identified
Data reveals the theory	Data is structured to reveal the theory
Coding is less rigorous. Take care not to 'over-conceptualize', identify key points	Coding is more rigorous. Codes are derived from 'micro-analysis' which consists of analysis of data word-by-word
Open, selective, and theorethical coding	Open, axial, selective, and 'coding for process'
Research questions should not be defined prior to data collection	Research questions may be defined upfront
An extensive literature review should be done after the theory is emerging	An extensive literature review is vital throughout the Grounded Theory processes

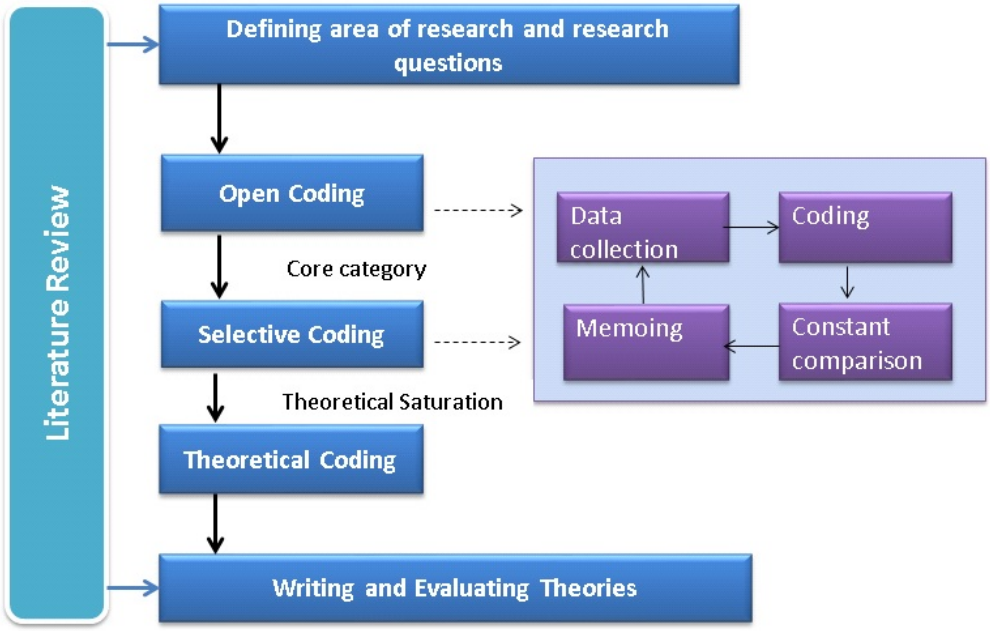


Figure 3.1: The Process of Grounded Theory of this Study Adapted from Hoda et al. [85]

the emerging theory is based solely on observed data, without imposing any preconceived ideas from the literature. This does not mean that the researcher has to start research without prior knowledge or ignore the literature [206]. The literature review facilitates the Grounded Theory researcher to identify the area of study which is required for further investigation [23, 203].

Conducting a minor literature review assists us as researchers to identify an area of interest. This study began by exploring Agile Software Development practices and its implementations. Based on the literature review, expertise coordination was identified as an area requiring further investigation and we have decided to explore this area in depth.

In order to explore the research area in depth, it is possible to formulate research questions. Formulating research questions, however, is not an integral part of Grounded Theory. The research question assists researchers to

determine suitable methods of collecting data [31] and detail the emerging theory [44]. Researchers should construct the research questions broadly, in order to provide freedom and flexibility for the researcher to explore the research area in depth [200, 203].

Prior to conducting data collection, we formulated three broad research questions. As findings of this study were emerging, the research questions, however, have been amended or changed in order to reflect the emergent concepts as follows:

- **How is expertise coordinated in Agile Software Development projects?** This question attempts to understand the expertise coordination process that should be carried out by Agile team members in managing expertise dependencies.
- **What are the strategies used in coordinating expertise outside Agile teams?** This question attempts to identify strategies used in coordinating expertise outside Agile teams.
- **What management strategies support expertise coordination in Agile Software Development projects?** This question attempts to determine the roles of management in supporting expertise coordination in Agile Software Development projects.

3.4.2 Data Collection

Glaser and Strauss [71] emphasize the importance of collecting data with multiple methods, which promise the construction of a novel theory. Although interviews appear to be the primary data collection method [23, 44], researchers can employ other methods. This study employed interviews as the predominant source of data collection, in conjunction with observations and document analysis.

Before commencing data collection, we applied for approval from the Human Ethics Committee (HEC) of Victoria University. We had applied for

HEC approval for conducting interviews (refer to Appendix A). The HEC application has been amended twice, in order to include observations and document analysis in conjunction with the interviews (refer to Appendices B and C).

The purpose of HEC approval is to protect the rights of participants and researchers [140]. As researchers, we need to be aware of the ethical issues, particularly confidentiality, privacy, and anonymity of participants. It is important to ensure that there is no identification of participants such as a participant's name, company, or the name of a specific project from the interview data. In order to protect the anonymity of the participants, we have assigned a number to each participant. As depicted in Table 3.2, each participant was labelled uniquely P1 to P48, in which letter 'P' stands for participant.

According to Miles and Huberman [140], as a part of ethical issues, informed consent is essential for gaining permission before conducting data collection. Therefore each participant was provided with an overview of this study and how to conduct the data collection. This information assisted them to provide written consent for their participation in this study.

Interviews

This study employed semi-structured interviews with open-ended questions. Interviews involve direct contact between researchers and participants during data collection. This situation enables the researcher to gain a deeper understanding of participants' concerns [31].

The participants for this study are Agile practitioners who engage in Agile Software Development projects. The Agile practitioners were recruited prior to conducting interviews. The process of recruiting participants and conducting interviews is described in the following subsections.

Recruiting Participants At the beginning, it was difficult to get Agile practitioners to participate in this study. We tried several ways to recruit participants. We started by searching for potential participants through recommendations from several former PhD students who were studying Agile Software Development. There was some frustration when one of the former PhD students mentioned that Agile practitioners were exhausted from many recent interviews by PhD students. This feedback, however, did not demotivate us to continue finding a way and opportunity to recruit participants for this study.

We started to recruit more participants after we joined the meetup group for the Agile Professionals Network branch in Wellington, New Zealand. Besides that, we used Agile conferences such as XP conferences, and Agile conferences based in New Zealand and the United States, as a platform to recruit more Agile practitioners. Attending Agile workshops, which were run by Agile training companies, was another avenue to identify and recruit potential participants in this study.

Semi-structured interviews were carried out with 48 Agile practitioners from different software organizations mainly based in New Zealand and the United States. The participants engage in different business domains such as education, finance, and human resources. This study was open to Agile practitioners who apply Agile practices in their software development projects even though there is no specific Agile method applied. Table 3.2 summarizes the details of participants involved in this study.

This study involved a broad range of Agile roles, such as developers, testers, and Agile coaches. This study also recruited external specialists such as Database Administrators and Architects, who are responsible for supporting Agile teams. In order to ensure coverage of the wide range of Agile roles, theoretical sampling was used by selecting subsequent participants for data collection based on existing data analysis [71]. Theoretical sampling ensures the comprehensive nature of data drawn from a broad range of participants. Different roles provide different insights and

perspectives of expertise coordination in the Agile Software Development context.

In Grounded Theory, there is no fixed sample size in recruiting participants [14]. The researcher continues to collect data until theoretical saturation has been reached, when no new data emerges [71]. Therefore, the number of participants depends on the emergent findings. In order to have wide coverage of phenomena studied, approximately 30 to 40 interviews are generally adequate to reach theoretical saturation [200]. We recruited 48 Agile practitioners, which is significantly more than the typical number of participants.

Conducting Interviews This study employed semi-structured interviews with a set of open-ended questions. These approaches provide rich data, which enabled us to gain insight into all aspects of the phenomena studied [142]. We constructed an interview guide, which consists of the following open-ended questions:

- The participant's background, experiences, roles, and responsibilities in Agile Software Development projects.
- An overview of Agile software project development projects that participants have been involved in.
- The participant's experiences in coordinating expertise in Agile Software Development projects.
- The participant's practices that are required for coordinating expertise in Agile Software Development projects.
- Challenges and barriers in coordinating expertise in Agile Software Development teams.
- Solutions to overcome the challenges and barriers in coordinating expertise in Agile Software Development teams.

Table 3.2: Summary of Research Participants and Agile Projects

Person	Location	Agile Role	Agile Methods	Project Domain	Project Duration	Team Size	Sprint
P1	New Zealand	Developer	XP and Scrum	Mobile application	2 months	10 to 20	1 week
P2	New Zealand	Agile Coach	XP, Scrum, Kanban	Not disclosed	Not disclosed	Not disclosed	Not disclosed
P3	Australia	Agile Consultant	Not disclosed	Not disclosed	Not disclosed	Not disclosed	Not disclosed
P4	New Zealand	Agile Coach	Scrum and XP	Education	1 year	6 to 7	2 weeks
P5	New Zealand	Software Tester	Not disclosed	Printing	2 months	20	2 weeks
P6	Australia	Team leader	Not disclosed	Accounting	3 months	7 to 10	2 weeks
P7	New Zealand	Agile Consultant	Scrum and XP	Financial	2 years	20	1 week
P8	Australia	Agile Coach	Scrum, XP, Kanban, Lean	Human resources	2 years	15	2 weeks
P9	New Zealand	Business Analyst	Not disclosed	Insurance	5 months	6 to 15	2 weeks
P10	New Zealand	Software Tester	Scrum	Education	1 year	9	2 weeks
P11	New Zealand	Project Manager	Scrum	Education	1 and half year	16	2 weeks
P12	New Zealand	Agile Coach	Scrum and Kanban	Not disclosed	Not disclosed	3 to 7	3 weeks
P13	New Zealand	Agile Coach	Scrum and Kanban	Government application	2 years	Not disclosed	2 weeks
P14	New Zealand	Product Owner	Not disclosed	Not disclosed	Not disclosed	Not disclosed	Not disclosed
P15	New Zealand	Agile Coach	Scrum and Kanban	Government application	Not disclosed	9	6 to 10
P16	New Zealand	Agile Coach	Scrum and Kanban	Government application	Not disclosed	6 to 7	2 weeks
P17	New Zealand	Tester	Scrum	Education	1 and half year	2-10	2 weeks
P18	New Zealand	Developer	Scrum	Education	1 and half year	16	2 weeks
P19	New Zealand	Business Analyst	Scrum	Education	1 and half year	16	2 weeks

Person	Location	Agile Role	Agile Methods	Project Domain	Project Duration	Team Size	Sprint
P20	New Zealand	User Experience Designer	Scrum	Not disclosed	less than a year	6 to 15	2 weeks
P21	New Zealand	Agile Coach	Scrum and Kanban	Mobile Application	Not disclosed	2 to 5	2 weeks
P22	New Zealand	Scrum Master	Scrum, Kanban, XP	Web-based Application	Not disclosed	6	2 weeks
P23	New Zealand	Developer	Scrum and XP	Dataware house	2 to 3 years	5 to 9	4 weeks
P24	New Zealand	Scrum Master	Scrum and Kanban	Banking	Not disclosed	5 to 13	2 weeks
P25	New Zealand	Developer	Scrum and Kanban	Financial	Nor disclosed	3 to 10	2 weeks
P26	New Zealand	Team Leader	Scrum and XP	Goverment Application	2 to 3 years	5 to 9	4 weeks
P27	New Zealand	Developer	Scrum and XP	Fishery	6 to 12 months	6	2 weeks
P28	Sweden	Developer	Kanban	Telecommunication	Not disclosed	5 to 13	2 weeks
P29	Denmark	Developer	Scrum	Medical	Not disclosed	10 to 12	4 weeks
P30	India	Business Analyst	Scrum and Kanban	Not disclosed	Not disclosed	15 to 20	2 weeks
P31	Malaysia	Scrum Master	Scrum and Kanban	Broadcast	4 months	3	3 weeks
P32	Malaysia	Scrum Master	Scrum, Kanban, XP	Enterprise	Not disclosed	5 to 6	1 weeks
P33	Malaysia	Project Manager	Scrum and Kanban	Security Application	1 to 2 years	7 to 8	2 weeks
P34	United States	Agile Coach	Scrum	Financial	3 months	6 to 7	2 weeks
P35	United States	Developer	Scrum	Financial	8 to 9 months	8 to 9	2 weeks
P36	United States	Developer	Scrum	E-commerce	1 to 6 months	6	2 weeks
P37	United States	DevOps	Not disclosed	Not disclosed	Not disclosed	Not disclosed	Not disclosed
P38	United States	User Experience Designer	Not disclosed	Not disclosed	Not disclosed	Not disclosed	Not disclosed

Person	Location	Agile Role	Agile Methods	Project Domain	Project Duration	Team Size	Sprint
P39	United States	Agile Coach	Scrum and XP	Not disclosed	Not disclosed	Not disclosed	2 weeks
P40	United States	Stakeholder	Not disclosed	Not disclosed	Not disclosed	Not disclosed	Not disclosed
P41	United States	Agile Coach	Scrum and XP	Biotechnology	Not disclosed	6 to 8	2 weeks
P42	United States	Tester	Scrum and XP	Retail	2 months to 2 years	4 to 11	2 weeks
P43	Wellington	DevOps	Scrum	Not disclosed	3 months	Not disclosed	4 weeks
P44	Wellington	Architect	Scrum	Oil Retail	3 to 4 months	Not disclosed	2 weeks
P45	Wellington	Tester	Scrum and Kanban	Financial	3 to 18 months	3 to 10	2 weeks
P46	Wellington	Agile Coach	Scrum, Kanban and XP	E-commerce	Not disclosed	7	2 weeks
P47	Wellington	Developer	Scrum, Kanban and XP	E-commerce	Not disclosed	7	2 weeks
P48	Wellington	Tester	Scrum and Kanban	Banking and Media	3 years	4 to 6	Not disclosed

Open-ended questions, however, require more depth and lengthier explanation from participants. In order to have direct contact with participants, we chose one-to-one interviews, instead of telephone or focus group interviews. One-to-one interviews require participants who are actively willing to speak and share their ideas [44].

In conducting the semi-structured interviews, sometimes the researcher needs to reorder or modify the open-ended questions for further investigation [142]. This situation aligns with the theoretical sampling in Grounded Theory, which requires the researcher to decide what data should be collected next based on the current findings [66]. Therefore, we kept changing the questions depending on data analysis.

At the beginning of this study, we recruited participants from Agile meetups. At the start of each interview, we asked a few questions to confirm and ensure the participants were implementing the right Agile practices. We abandoned one interview after discovering the participant did not implement the right Agile practices.

Then, we asked general questions related to the topic of our study as stated in the interview guidelines. The open-ended questions work as a guideline and we were not restricted and rigid with the questions. During the interviews, we tried to adapt to participants' ideas and thoughts, and kept asking questions to gain in-depth information. As this study progressed and required further investigation, we started to recruit Agile practitioners with specific Agile roles, including external specialists. At this stage, we asked more specific questions, and focused more on important aspects that emerged from our findings.

Each interview took approximately 45 to 60 minutes to complete. As the interviews were extremely time intensive, we used an audio recorder to record the interview data. Glaser [66] argues that recording is time consuming because the researcher tends to transcribe and analyze non-important parts of the data. In this study, however, we chose to record interviews to ensure no data was missed.

After the interviews, some participants were contacted through email or phone for further clarification on unclear, incomplete, and ambiguous interview data. We also obtained their feedback by forwarding our published conference papers that contained their interview quotes. These activities help establishing correctness of data interpretation [31].

Observations and Document Analysis

This study employed observations and document analysis in conjunction with interviews. Observations and document analysis are classified as secondary data collection methods of this study. The main purpose of these methods is to confirm the accuracy of data interviews and enhance the validity of data [96].

Observations Observations provided a great opportunity to view the actual participants' behaviour when they were engaging in Agile Software Development projects. Moreover, observations allowed us to gain a deeper understanding of the participants' settings [98, 165]. The advantages of observations led us to identify new findings and also enable data triangulation.

At the beginning of this study, we intended to carry out observations at least at three different Agile Software Development companies. As it was difficult and challenging to get an Agile Software Development company to participate in this study, we managed to carry out observations at one company only, based in Wellington. Therefore, the observations were conducted after all interviews had been done.

In this thesis, we refer to the company as 'company XYZ' in order to protect the anonymity of the company. We had about six observational sessions over two months, observing two different Agile teams, who worked on different software development projects as summarized in Table 3.3.

Creswell [44] lists four types of observation engagement: *complete participant*, *participant as observer*, *nonparticipant/observer as participant*, and

Table 3.3: Observations.

Team	Project Domain	Team Size	Project Duration	Session
A	Metadata search tool and various front end applications	4 developers 1 scrum master	5 years	Stand-up meeting Sprint planning Sprint review Sprint retrospective
B	New corporate website	2 developers, 1 designer 1 scrum master	12 weeks	Stand-up meeting Sprint planning

complete observer. The observations of this study can be categorized as *nonparticipant/observer as participant*, where we were involved as outsiders without direct involvement in activities or participants. This enabled us to watch and take notes from a distance.

Observations required us to pay attention to participants, physical settings, interactions, and activities [44]. As this study focuses on expertise coordination, we made it a priority to observe how Agile team members interacted with each other in managing their expertise resources. We used Agile meetings such as daily stand-up meetings, sprint planning, and retrospectives to observe teams' interactions and behaviours when locating, recognizing, and accessing expertise in teams.

Observations were recorded in the form of observation notes. During observations, we wrote down important incidents, behaviours, and experiences that related to phenomena studied. We developed full observation notes once integrated with our personal reflections, insights, and ideas.

Document Analysis

Document analysis is another alternative for Grounded Theory researchers to collect and elicit more data [31, 71]. Document analysis involves a process of reviewing or evaluating printed or electronic documents, that have been produced without researchers' intervention [25]. Charmaz [31] identifies two different types of documents involved in providing enriched data into Grounded Theory study:

- Elicited document - requires participants to produce written materials such as daily logs, diaries, and journals in order to convey their views in the phenomena studied.
- Extant document - requires researchers to review existing documents such as meeting minutes, pictures, diagrams, and screen interfaces.

Extant documents were suitable for this study since we gained access to supporting documents such as Agile artefacts, management documents, screen interfaces, and pictures. Document analysis served as a complement to interviews and observations in order to support and confirm the interviews and observations data. We collected relevant documents during interviews and observation, including:

- Pictures of Agile meetings such as daily stand-up, sprint planning, retrospective and sprint review meetings.
- Pictures of Agile working spaces.
- A picture of tasks and role dependencies.
- A performance appraisal form.
- Screenshots of Wikis.

During analysis, we looked for consistency between interviews, documents, and observations. Sometimes, document analysis sparked new ideas that led to the emergence of new findings and further data collection.

3.4.3 Data Analysis

The difference between Grounded Theory and other qualitative methods is the continuous interplay between data collection and data analysis [42]. Glaser argues that separating data collection and data analysis prevents the emergence of theory [71]. Simultaneous data collection and analysis assisted us to find gaps or holes in data, and enabled us to go further and deeper in our phenomena studied. Thus, data analysis began as soon as the first interview had been carried out.

The term 'coding' is associated with Grounded Theory data analysis. Charmaz [31] defines coding as a pivotal link between collecting data and developing an emergent theory. Coding enables researchers to fracture data, group it into codes, and develop the hypothetical relationships between categories [66]. In this thesis, we use several terms that are related to the coding process. Table 3.4 depicts a glossary of coding terms that apply in our data analysis.

In order to understand the coding process, it is necessary for researchers to have a clear picture of abstraction levels of Grounded Theory. Figure 3.2 adapted from Hoda [83] summarizes the level of abstraction in Grounded Theory.

In this study, we employed three stages of coding proposed by Glaser and Corbin [71]: open coding, selective coding, and theoretical coding. Open coding and selective coding intend to produce substantive codes. Substantive codes are the emergent categories and properties that conceptualize the phenomena studied [66]. The purpose of theoretical coding is to generate theoretical codes, which conceptualize interrelation between substantive codes as hypotheses or emergent theory [66]. The following subsections describe each coding stage in detail.

Table 3.4: A Glossary of Coding Terminologies.

Terminology	**Definition**
Data	Raw material collected through data collection such as interviews.
Key point	Relevant words or phrases that represent the concern of participants.
Code	A form of shorthand that researchers repeatedly use to identify conceptual re-occurrences and similarities in the patterns of participants' experiences.
Concept	Collections of codes of similar content that allow the data to be grouped.
Category	A group of similar concepts that are used to generate a theory.
Core category	The main theme or concern of participants.
Properties	Characteristics that are common to all the concepts in a category.
Theory	An explanatory scheme comprising a set of concepts related to each other through logical patterns of connectivity.
Memoing	A fundamental analytical process in Grounded Theory research that involves the recording of processes, thoughts, feelings, ideas, and decisions in relation to a piece of research.
Constant Comparison	Data is compared with all existing concepts to form a category or to point to a new relation.

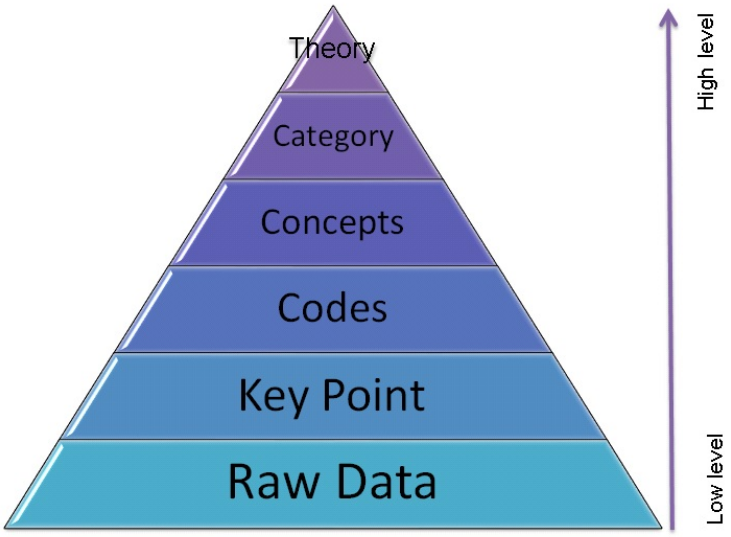


Figure 3.2: Levels of Abstraction in Grounded Theory (Adapted from Hoda [83]).

3.4.3.1 Open Coding

The aim of open coding is to generate an emergent set of concepts and properties, which leads to emergent categories [66]. Open coding required us to remain open without preconceived ideas in exploring whatever possibilities lead to the emergence of theory. During this process, we had to understand the data by attempting to answer the following questions proposed by Glaser [66]:

- "What is this study of?"
- "What category does this incident indicate?"
- "What is actually happening in the data?"
- "From whose point of view?"

Open coding involves a process of examining the data, breaking up the data into segments and then attaching a label for each segment [200]. As line-by-line coding is a time-consuming and painstaking effort [66], we decided to use key point coding. We examined words, phrases, and sentences in order to collate key points [7]. Then we rephrased the key point with a meaningful label, which led to the formation of a code [7]. This step of coding excited the researchers because digging out new codes led to rich research findings.

Charmaz [31] suggests researchers identify events involved and code the events as actions. Therefore, we named the code with a gerund to represent the actions involved in this study. For example:

> **Interview Transcript:** *"We realized that we have many projects coming. So, it is important to get alignment between the project and people. When we started, we had 110 people, and now 200. It becomes harder to keep track of a lot of skills. So, this tool helps me to identify the skills."*- P11, Project Manager.
>
> **Key point:** this tool helps me to identify the skills
>
> **Code:** identifying expertise through an expertise directory

Many key points were generated through open coding. In order to easily distinguish and trace each key point, we used a unique identifier to represent each code as depicted in Figure 3.3. KP2 1 represents the first key point which belongs to Participant P2. Table 3.5 shows how we applied the unique identifier for each key point that led to the emergence of codes for Participant P16.

Constant Comparison In order to look for commonalities and differences, we used constant comparison to compare codes and codes, codes and concepts, and concepts and concepts [69]. Constant comparison is a central

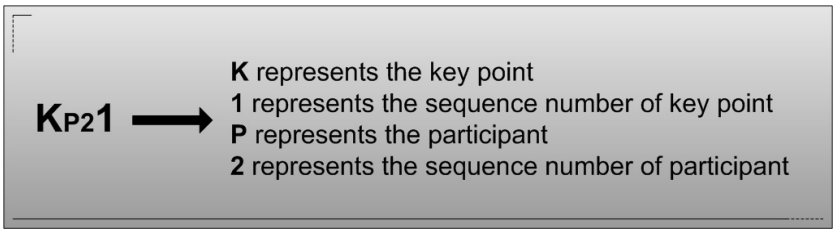


Figure 3.3: Identifier Format for Representing Key Points.

process of Grounded Theory, which purposely intends to identify patterns and variations of codes, concepts, and categories [31, 203]. Through constant comparison, existing concepts, and categories can be enhanced or new concepts or categories formed [214].

We first used constant comparison to compare codes with one another for establishing uniformity between codes. Similar codes with common themes were grouped together and emerged as a concept. Table 3.6 depicts an example of codes that generated the concept *using an expertise directory* by using constant comparisons.

Many concepts emerged, and repeated constant comparison was undertaken by comparing new codes with emergent concepts. The challenge of constant comparison began at this point when the number of codes and concepts increased. Comparing each new code to all the existing codes required a supporting tool to organize the data. We decided to use NVivo as a data analysis tool to facilitate the process of constant comparison (see section 3.4.3.4).

Then, the process of constant comparison continued by comparing between concepts, and led to the emergence of categories. A category is a group of similar concepts that are used to generate a theory [71]. Eleven categories emerged from our data analysis:

- *locating and recognizing expertise*
- *accessing expertise*

Table 3.5: Example of Key Points and Codes Extracted from Participant P16

ID	Key Point	Code
K P16 162	They are filling the gap, supporting each other and understanding other roles	Filling other roles
K P16 166	We don't know the people who will be in the project	Having no idea of team member's expertise
K P16 167	We can work well together and sit together, and we have to make sure we get the right people as a team	Having the right team members
K P16 171	In skill matrix, how do you write yourself, in terms of number one or number two to number five	Updating an expertise directory
K P16 174	I would not rather have someone who is absolutely fantastic in Java but they didn't communicate or they didn't care about testing	Preferring people skills over technical skills
K P16 175	A capacity in there in picking up a new skill or a capacity there for picking up a new pattern	Learning continuously
K P16 177	We kicked off the release planning inception meeting. We invited everybody. It is about the setting the vision, high level scope, and etc	Engaging all together at the beginning of project
K P16 179	They don't have the project's visibility. They didn't know when they are needed	Having ineffective communication
K P16 189	If we get somebody else in, how could we extract the knowledge and learn from them? The skills set needs to be sustainable	Sustaining outside expertise
K P16 191	They tried to get as many easy user stories as possible as they can, in order to make the matrix look good. It doesn't force the whole team spirit	Gaming the system to fulfil an individual KPI

Table 3.6: An Example of the Concept *Using an Expertise Directory*.

Codes	**Sources**
Gaining benefits through an expertise directory	P11, P18, P33
Updating the skill set	P2, P11, P16, P17, P18, P31, P33
Preferring other technique	P2, P39
Depending on organization's needs	P11, P33

- *planning ahead*
- *understanding Agile mindset*
- *ensuring consistency*
- *retaining external expertise*
- *keeping everyone on the same page*
- *treating outsourcers the same as in-house staff*
- *self-selecting teams*
- *reforming performance appraisal*
- *embracing an expertise sharing culture*

Each category consists of concepts that relate to a common theme. For example, the category *locating and recognizing expertise* describes how Agile teams identify who has which sort of expertise, and when and who needs the expertise. Figure 3.4 shows how the category was formed based on underlying concepts: *communicating frequently*, *working closely together*, *observing expertise*, *declaring self-identified expertise*, *earning certification*, and *using an expertise directory.*

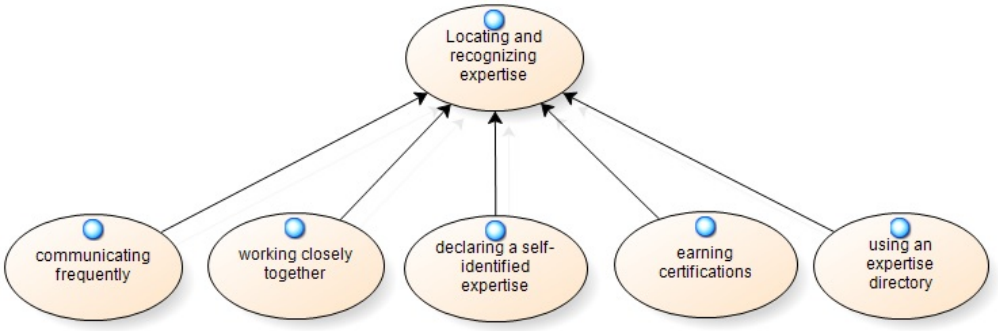


Figure 3.4: The Emergence of the Category *Locating and Recognizing Expertise* from the Underlying Concepts.

As we proceeded with constant comparison, a core category started to emerge. A core category is the highest level of data abstraction in Grounded Theory, which represents the main theme or concern of participants [66]. Glaser [66] argues that the core category must be central and related to many other categories and their properties. The core category is selected based on its frequent re-occurrence during coding.

Based on Glaser's criteria in determining the core category [66], *locating and recognizing expertise* appeared to subsequently emerge as the core category. The core category *locating and recognizing expertise* is central and related to other categories, and has been discussed thoroughly in the theoretical coding subsection (refer to subsection 3.4.3.3).

Memoing It is essential for Grounded Theory researchers to engage with memoing constantly through data analysis. Memoing is a core step of Grounded Theory that assists researchers in developing ideas. Glaser [66] defines memos as "*the theorizing write-up of ideas about codes and their relationships as they strike the analyst while coding*". Memos consist of the researcher's ideas that lead to developing categories necessary for the construction of theory. Memoing enables researchers to articulate the following aspects of theory development [66]:

- "It raises the data to a conceptualization level."
- "It develops the properties of each category which begin to define it operationally."
- "It presents hypotheses about connections between categories and/or their properties."
- "It begins to integrate these connections with clusters of other categories to generate the theory."
- "It begins to locate the emerging theory with other theories with potentially more or less relevance."

Birks and Mills [23] list items that researchers can include in their memos:

- The researcher's feelings and assumptions about research.
- The researcher's philosophy and position in relation to research.
- Important points from papers, books, or any reading material.
- Potential problems or issues in relation to research.
- Codes, categories, and developing theories.

Memoing is a natural and spontaneous process without scheduling [66]. There is no rule on how to write a memo. It is up to researchers to put their ideas into memo without considering its presentation. Proper sentence construction is irrelevant when memoing.

We found memoing was not easy and required us to keep capturing thoughts, feelings, and analytic ideas related to the data. This process continued through the entire data analysis process. It was essential for us to keep up the momentum of writing memos.

Constant comparison leads to the need of memoing [66]. The outcome of constant comparison needs to be recorded and integrated with researchers'

ideas. Therefore, memoing is constantly continued at every stage of coding until the emergence of theory. This requires researchers to keep memoing and allows modification and refinement of memos.

We started memoing at this stage of coding purposely to integrate the constant comparison outcome with our ideas. Memoing enabled us to see the progress development of theory, the gaps in existing data analysis and new possible directions for the emerging theory. Figure 3.5 shows an extract of a concept memo for *using an expertise directory*.

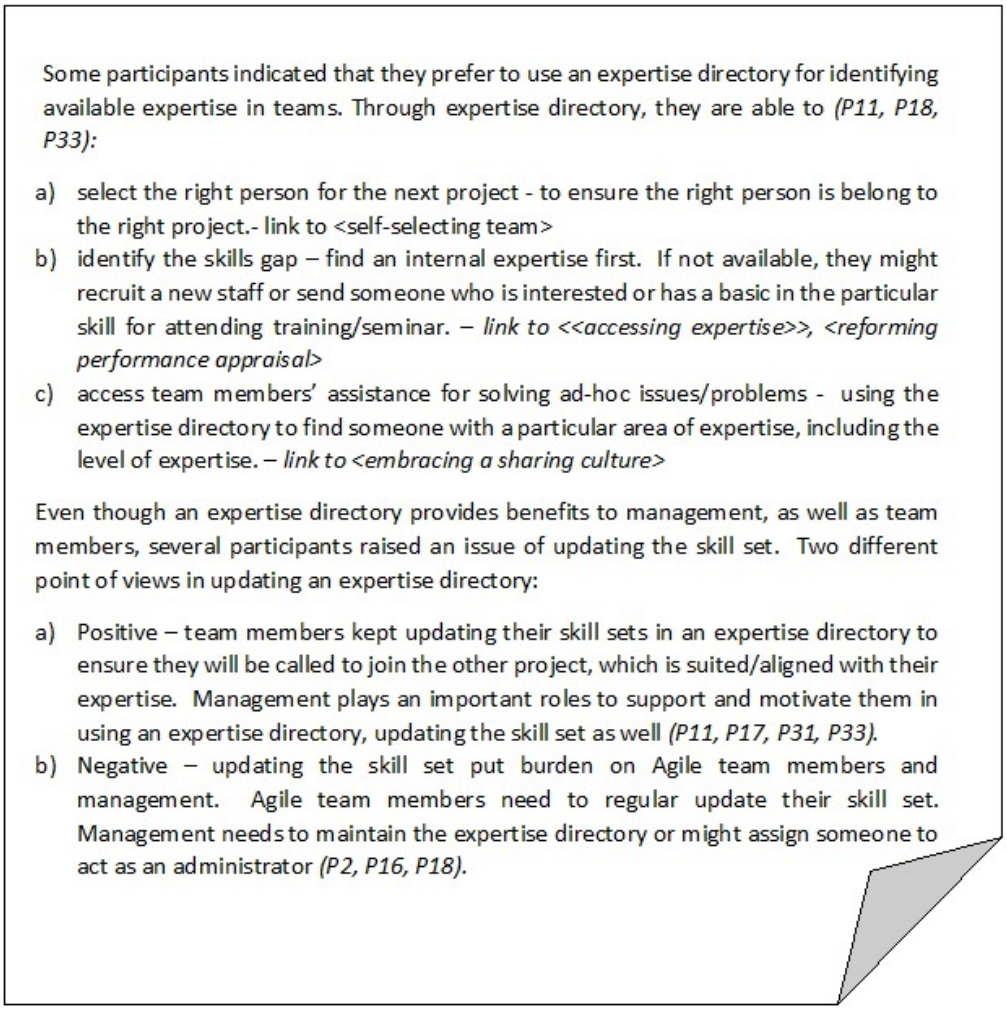
3.4.3.2 Selective Coding

Researchers move to the next step, selective coding, after the core category has emerged [66, 67]. At this stage, the core category acts as a baseline for further data collection and analysis, as well as theoretical sampling [66, 67]. Once the core category *locating and recognizing expertise* emerged, we analysed data based on the emergent core category, which was coded selectively on categories that were related to the core category.

Memoing for this stage of coding became more focused and reflected the core categories [66]. Hence, memoing led researchers to identify gaps for theoretical sampling, which would probably require new data. Collecting data, however, must be conducted within the core categories context. We continued collecting data until no new data emerged, i.e until theoretical saturation.

3.4.3.3 Theoretical Coding

Theoretical coding is the final stage of coding, after theoretical saturation. Glaser [66] defines theoretical coding as *"a property of coding and constant comparative analysis that yields the conceptual relationship between categories and their properties as they emerge".* Theoretical coding generates theoretical codes, which conceptualize how substantive codes are related to each other as hypotheses, and then are integrated into a theory.



Some participants indicated that they prefer to use an expertise directory for identifying available expertise in teams. Through expertise directory, they are able to *(P11, P18, P33):*

a) select the right person for the next project - to ensure the right person is belong to the right project.- link to <self-selecting team>
b) identify the skills gap – find an internal expertise first. If not available, they might recruit a new staff or send someone who is interested or has a basic in the particular skill for attending training/seminar. – *link to <<accessing expertise>>, <reforming performance appraisal>*
c) access team members' assistance for solving ad-hoc issues/problems - using the expertise directory to find someone with a particular area of expertise, including the level of expertise. – *link to <embracing a sharing culture>*

Even though an expertise directory provides benefits to management, as well as team members, several participants raised an issue of updating the skill set. Two different point of views in updating an expertise directory:

a) Positive – team members kept updating their skill sets in an expertise directory to ensure they will be called to join the other project, which is suited/aligned with their expertise. Management plays an important roles to support and motivate them in using an expertise directory, updating the skill set as well *(P11, P17, P31, P33).*
b) Negative – updating the skill set put burden on Agile team members and management. Agile team members need to regular update their skill set. Management needs to maintain the expertise directory or might assign someone to act as an administrator *(P2, P16, P18).*

Figure 3.5: An Extract from a Memo of Concept *Using an Expertise Directory*.

A theoretical code is a relational model which consists of all substantive codes (categories) related to the core category [80]. Through theoretical coding, we can determine the relationship between categories, and the core category as well. These relationships put the 'fractured' substantive codes together into a theory. The theoretical codes impose a framework of analysis within a new insight and broad picture [66]. Hence, theoretical

coding enhances the abstraction of data, which enables the data analysis to be more precise, coherent, and comprehensible [31].

In order to integrate substantive codes, Glaser [66] defines 18 theoretical coding families, which indicate specific distinct concepts. For example, the strategy family includes several concepts such as *tactics*, *mechanisms*, *goals*, and *techniques*. Glaser [70, 203] then enlarges several theoretical coding families and enhances the list of concepts.

At this stage of coding, sorting memos is essential. Sorting memos provides a visual insight into theoretical coding identifying logical interactions between substantive codes. The sorting process, however, should involve conceptual sorting instead of data sorting [66]. That is important to deeply understand all categories and have sensitivity in determining the relationships between categories. This leads to the emergent theoretical code and theory formulation, and ultimately assists in writing theory.

Sorting memos was done in this study by using the by-hand method instead of computer software program. Based on Glaser's rules of memoing [66], we wrote memos on index cards with proper labels. We sorted the memos on a table by considering similarities, connections, and conceptual orderings. In order to clearly see the conceptual sorting, we constructed a diagram to visualize the sequencing of conceptual categories. According to Charmaz [31], drawing a conceptual diagram assists researchers to see the scope and direction of the categories as well as to grasp better connections between categories.

Eleven substantive codes or categories emerged from our data analysis. At the initial stage of theoretical coding, we determined the relationships between categories, and between the core category and categories. Figure 3.6 shows the emergence of initial theoretical code with nine categories. The dashed arrow represents a relationship between categories within the same core research findings, whereas the bold arrow indicates the relationship between categories from different core research findings.

The initial theoretical code was refined several times to ensure the

direction of each category and connection between the categories. We also refined the theoretical code based on Glaser's theoretical coding family as depicted in Figure 3.7. The theoretical codes are described thoroughly in Chapter 4. Two theoretical coding families fit these substantive codes as follows:

- *Process family* for substantive codes *locating and recognizing expertise* and *accessing expertise*.
- *Strategy family* for substantive codes *coordinating outside expertise, self-selecting teams, reforming performance appraisal,* and *embracing an expertise sharing culture.*

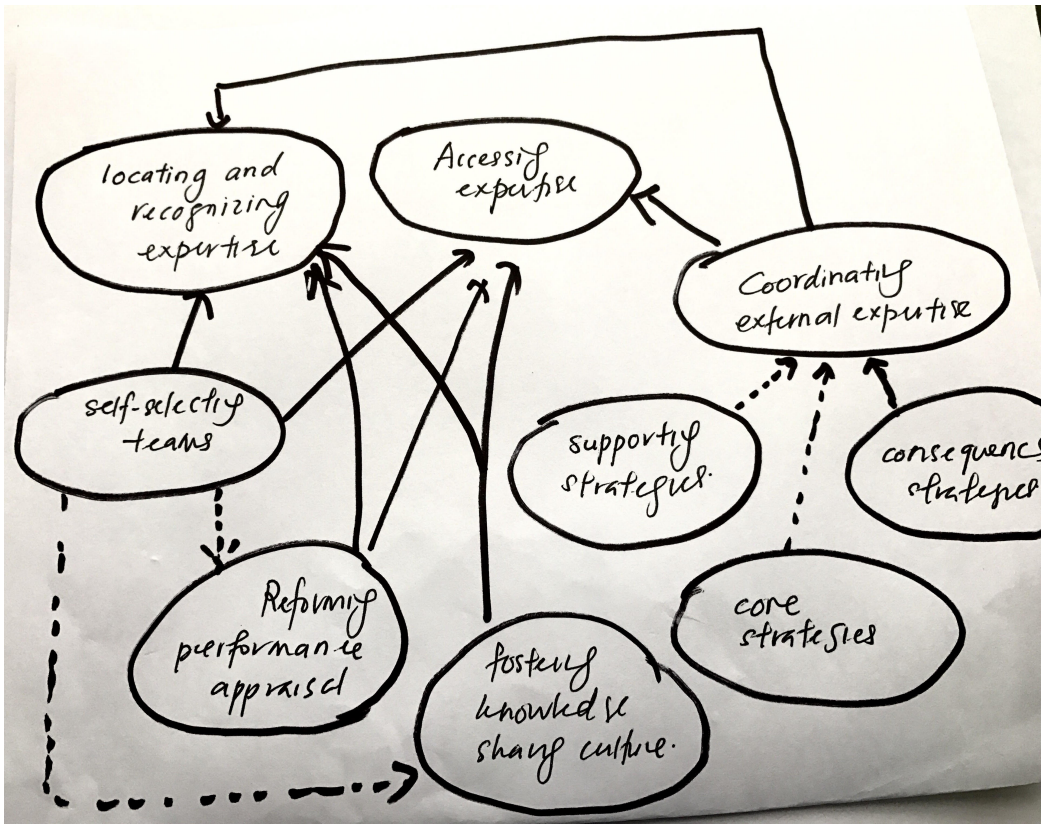


Figure 3.6: The Initial Theoretical Code of this Study.

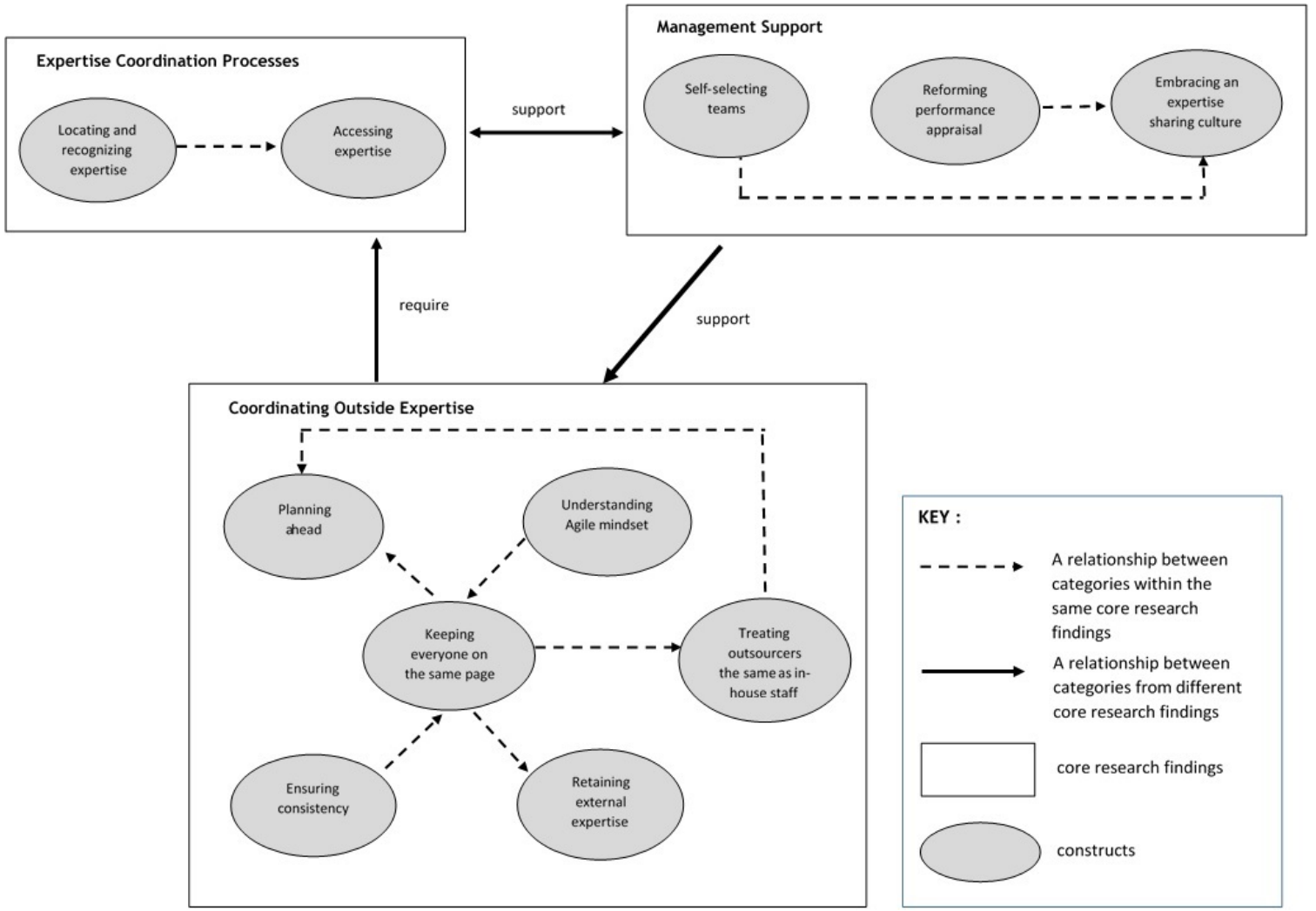


Figure 3.7: The Emergent Theoretical Code of this Study.

3.4.3.4 Data Analysis Tools

We used a combination of both manual and computer-based tools in the data analysis process. NVivo was chosen over other data analysis software because it is relatively simple to use [114]. The use of NVivo is not to replace our roles as researchers in analysing the data, but it assists and enhances the data analysis process by:

- recording, storing, indexing, sorting, and coding data
- facilitating a constant comparison process

We imported interview transcripts into NVivo, which enabled us to code directly. NVivo facilitated coding process through the creation of nodes. As the number of nodes increased, NVivo allowed us to create tree nodes in order to group nodes as concepts or categories. This process facilitated us to store and organize our data effectively, as well as conduct constant comparison. Figure 3.8 depicts the some of the tree nodes that we used to organize the emergent concepts and categories of this study.

There are no specific guidelines on how to conduct qualitative data analysis using NVivo [114]. Based on NVivo tutorials that were available on the Internet, we explored how to use the software to store, organize, and analyse the data. As we were unable to attend a hands-on NVivo workshop, this limited the use of the NVivo functions. We managed to use NVivo for analysing data through:

- coding stripe - to view nodes that are coded to a specific text or document.
- query - to search a specific word or phrase in order to understand participants' concerns.
- model - to present nodes and relationships diagrammatically.
- report - to summarize specific information of interest that we sought for more understanding.

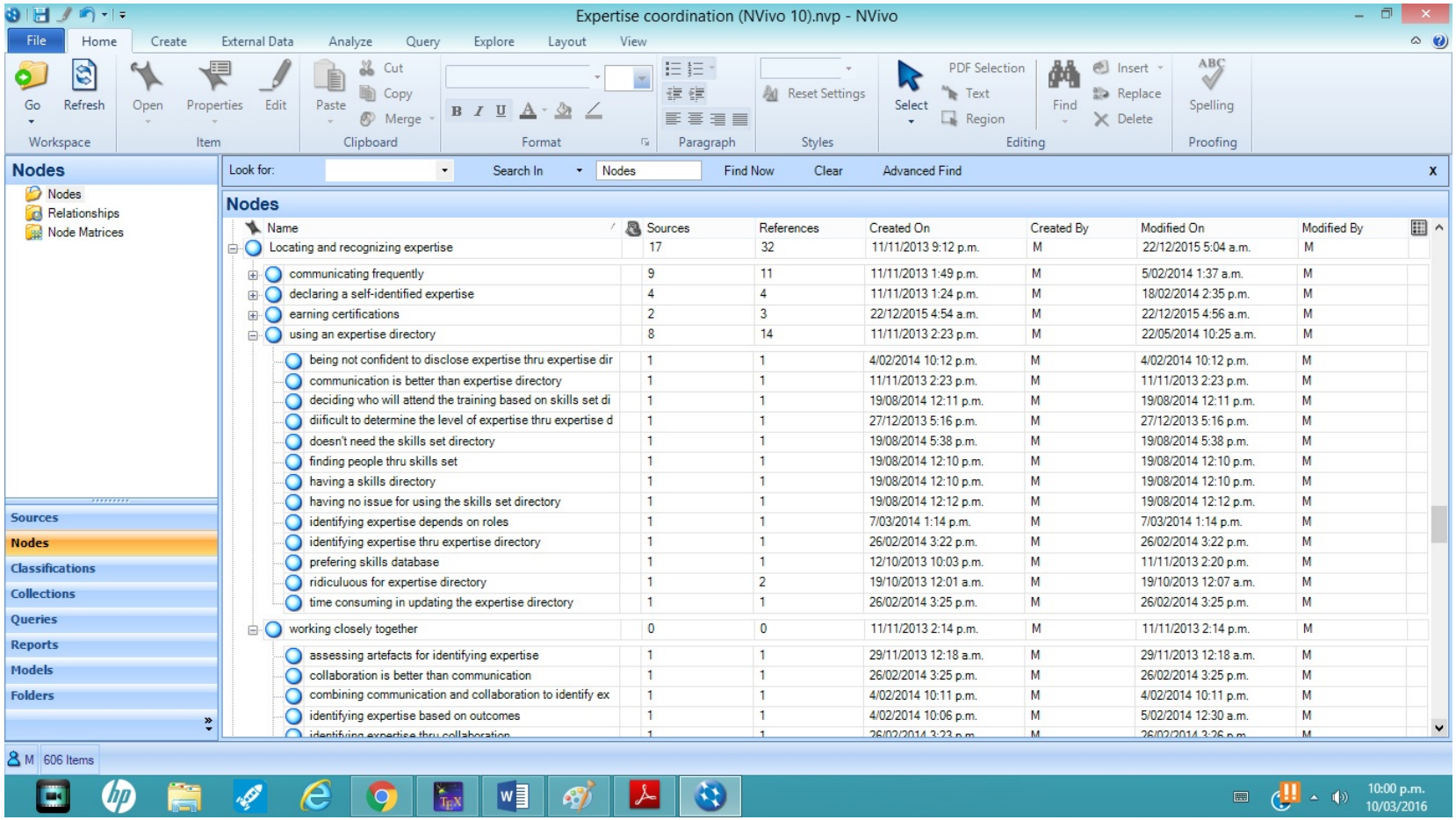


Figure 3.8: The Example of Tree Nodes in NVivo.

Besides NVivo, we also used manual methods such as sticky notes, sketching, and index cards during conceptual analysis. These methods were effective for sorting and linking categories, which provided visibility of the categories and its relationships.

3.4.4 Generating a Theory

Strauss and Corbin [203] define a theory as *"a set of well-developed categories that are systematically inter-related through statements of relationships to form a theoretical framework that explain some relevant social, psychological, educational, nursing, or other phenomenon."* The theory is an integrated set of hypotheses, which provides an explanation of phenomena [71, 203]. Weber [225] describes a theory as a conceptual model that represents some subset of phenomena studied. The theory of our study is a set of categories and their relationships that describes expertise coordination in Agile Software Development projects.

Two types of theory can be generated in Grounded Theory: substantive theory and formal theory [71]. Substantive theory is generated to address a specific area of phenomena. A constant comparison among different substantive theories leads to the development of formal theory. A formal theory is abstract, general, and the highest level of Grounded Theory. Figure 3.9 depicts the different levels of theories in Grounded Theory which are adapted from Urguhart [215]. A substantive theory has been generated in this study since our study focuses on a specific area of phenomena, expertise coordination in Agile Software Development projects.

Some scholars define different essential components of theory. For instance, Whetten [230] proposes the following theory components:

- *"What factors (variables, constructs, or concepts) should be considered as a part of phenomena studied?"*
- *"How the factors are related?"*

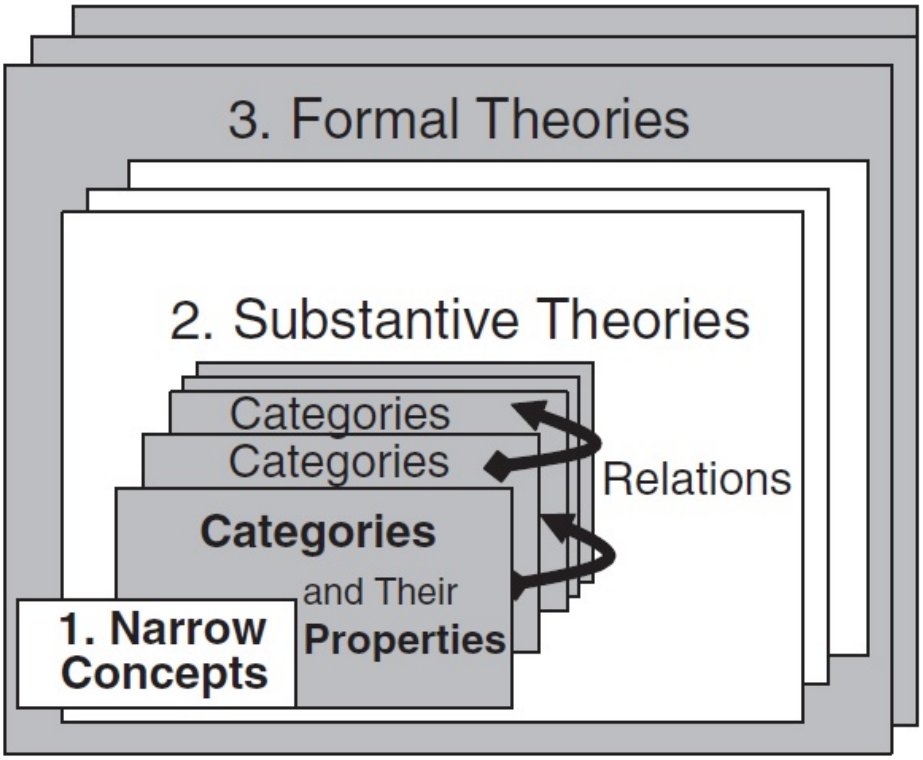


Figure 3.9: The Level of Theory in Grounded Theory (Adapted from Urguhart [215]).

- *"Why the factors and their relationships are important in representing the phenomena studied?"*
- *"Who, where, and when to generalize the factors and relationships in other possible settings of study?"*

In Information Systems, Weber [225] defines four components of theory as follows:

- Constructs - to represent an attribute or class of things in its domain.
- Associations - to show the value of construct is related to the other value of construct.

- States - to represent a state of attributes of things.
- Events - to represent a change from one of its states to another of its states.

Weber [225] states that the components of theory should be defined precisely, because they define the theory boundaries. We describe the theory boundaries in Chapter 4.

Both scholars [225, 230] indicate that constructs and associations are prevalent in generating a theory. Constructs are defined in this study through the emergent categories and concepts. The detailed explanation of our theory's constructs and its relationships are described in detail from Chapters 4 to 8.

3.4.5 Evaluating the Theory

This study employed multiple approaches for evaluating the emergent theory:

- Lincoln and Guba's criteria
- Glaser's criteria
- Weber's criteria

Lincoln and Guba's criteria [74] was used as a threat to validity of this study. As depicted in Table 3.7, this study considered four criteria for evaluating our study as a qualitative study. Each criteria has specific approaches for ensuring the trustworthiness of this study. For instance, credibility was examined through prolonged engagement, persistent observation, triangulation of data, peer debriefing, negative case analysis, and member checks.

Specific to Grounded Theory, we evaluated our study according to Glaser's criteria [66, 67], as depicted in Table 3.8. Besides Glaser's criteria, this study has also evaluated the quality of emergent theory based on

Table 3.7: Lincoln and Guba's Criteria Adapted from Lincoln and Guba [74].

Criteria	**Purpose**
Credibility	to ensure the truth value of findings
Transferability	to ensure the applicability of this study to be applied in other contexts
Dependability	to determine the consistency of research process, which leads to the stability of the data over time
Confirmability	to avoid bias issues and prejudices of the researchers

Table 3.8: Glaser's Criteria Adapted from Glaser [66, 67].

Criteria	**Purpose**
Fit	to ensure our theory has emerged naturally from data without forcing the data to fit preconceived ideas
Work	to ensure our theory can represent the participants' concerns and demonstrate the application value
Relevance	to ensure the ability of the emergent theory to be related to core problems and processes in the area under study
Modifiability	to ensure the emergent theory is flexible to adapt to modifications in the future

Weber's five criteria: importance, novelty, parsimony, level, and falsifiability [225]. Table 3.9 depicts the list of Weber's criteria, and a detailed explanation of each evaluation criteria is explained in Chapter 8.

Table 3.9: Weber's Criteria Adapted from Weber [225].

Criteria	**Purpose**
Importance	to determine the importance of the emergent theory from the viewpoint of practice and research
Novelty	to determine the value ascribed to it by researchers and the publication acceptance for describing the emergent theory
Parsimony	to ensure a simple theory is created with a small number of categories and relationships
Level	to determine the level of emergent theory: micro-level, middle-range level, or macro-level
Falsifiability	to ensure a high-quality theory might be falsified via empirical tests

3.5 The Roles of the Researcher

The researcher is an integral component of Grounded Theory, and contributes to the accuracy of data interpretation. Therefore, self-reflection of the researcher is essential to successful Grounded Theory research. The following statement describes the researcher's background and how it contributes to the researcher's roles:

I completed a Bachelor of Information Technology from Universiti Utara Malaysia, Malaysia in 2000. After finishing the bachelor's degree, I worked as a software developer. Through several software projects, I obtained hands-on experience and a good understanding of software development practices.

After two years, I changed the direction of my career into the academic world. Working as a tutor urged me to undertake a Master of Computer

Science (Software Engineering) in the Centre for Advanced Software Engineering (CASE), Universiti Teknologi Malaysia. A nine month internship was a requirement to obtain the Masters degree, which gave me the opportunity to explore mobile application development in an Object-Oriented programming environment.

Since finishing my Masters, I have been working as a lecturer at Universiti Utara Malaysia. As research is a component of performance appraisal, I have been involved in several different funded research projects including conceptual model development and survey research. The experience that I gained through these research projects taught me about the research process and facilitated me in conducting this study.

Teaching software engineering courses such as Agile Software Development enabled me to explore the theory of Agile methods in depth. In order to strengthen my knowledge and understanding in Agile methods, I was involved in two research projects that were related to the Agile Software Development context. These experiences enabled me to determine the research area in Agile Software Development that needed further investigation.

4

A Theory of Expertise Coordination

This chapter introduces a theory of expertise coordination in the context of Agile Software Development Projects. This chapter describes the theory's presentation, and provides an overview of its emergent categories and relationships. The last section of this chapter presents the boundaries of the theory.

4.1 The Theory's Presentation

The theory is represented in a model that consists of hypotheses that describe expertise coordination in Agile Software Development projects. As previously mentioned in section 3.4.4, a theory should consist of constructs and associations. Constructs are defined in this study through the emergent categories and concepts, whereas associations are represented in our theory through the relationships between categories.

Figure 4.1 shows the theory of expertise coordination for Agile Software Development Projects. The rectangles represent the three core research findings that emerged from our data analysis: *expertise coordination process*, *coordinating outside expertise* and *management support.* Each core research finding consists of several categories, which are represented as ovals. The figure depicting the *expertise coordination process* consists of two categories: *locating and recognizing expertise* and *accessing expertise. Management support* consists of three categories: *self-selecting teams*, *reforming performance appraisal* and *embracing an expertise sharing culture. Coordinating outside expertise* comprises six categories: *planning ahead*, *understanding Agile mindset*, *ensuring consistency*, *retaining external expertise*, *treating outsourcers the same as in-house staff*, and *keeping everyone on the same page*.

The emergent theory of this study includes several binary relationships, which indicate associations between two categories. The dashed arrow represents a relationship between categories within the same core research finding. For instance, *expertise coordination process* comprises two sequential steps: *locating and recognizing expertise* and *accessing expertise.* The bold arrows summarize the relationships between different categories from core findings. For instance, *management support* plays an important role to support the *expertise coordination process*, as well as *coordinating outside expertise.* Successful *coordinating outside expertise* relies on *expertise coordination process.* The following subsections provide a detailed explanation of each category and their relationships.

4.2 Expertise Coordination Process

Expertise coordination requires a team to recognize who has particular expertise, when and where they are needed, and how to access the expertise effectively. Therefore, it is important to identify who has particular expertise and to recognize the need for the expertise. The identified expertise then can be accessed when needed in a timely manner.

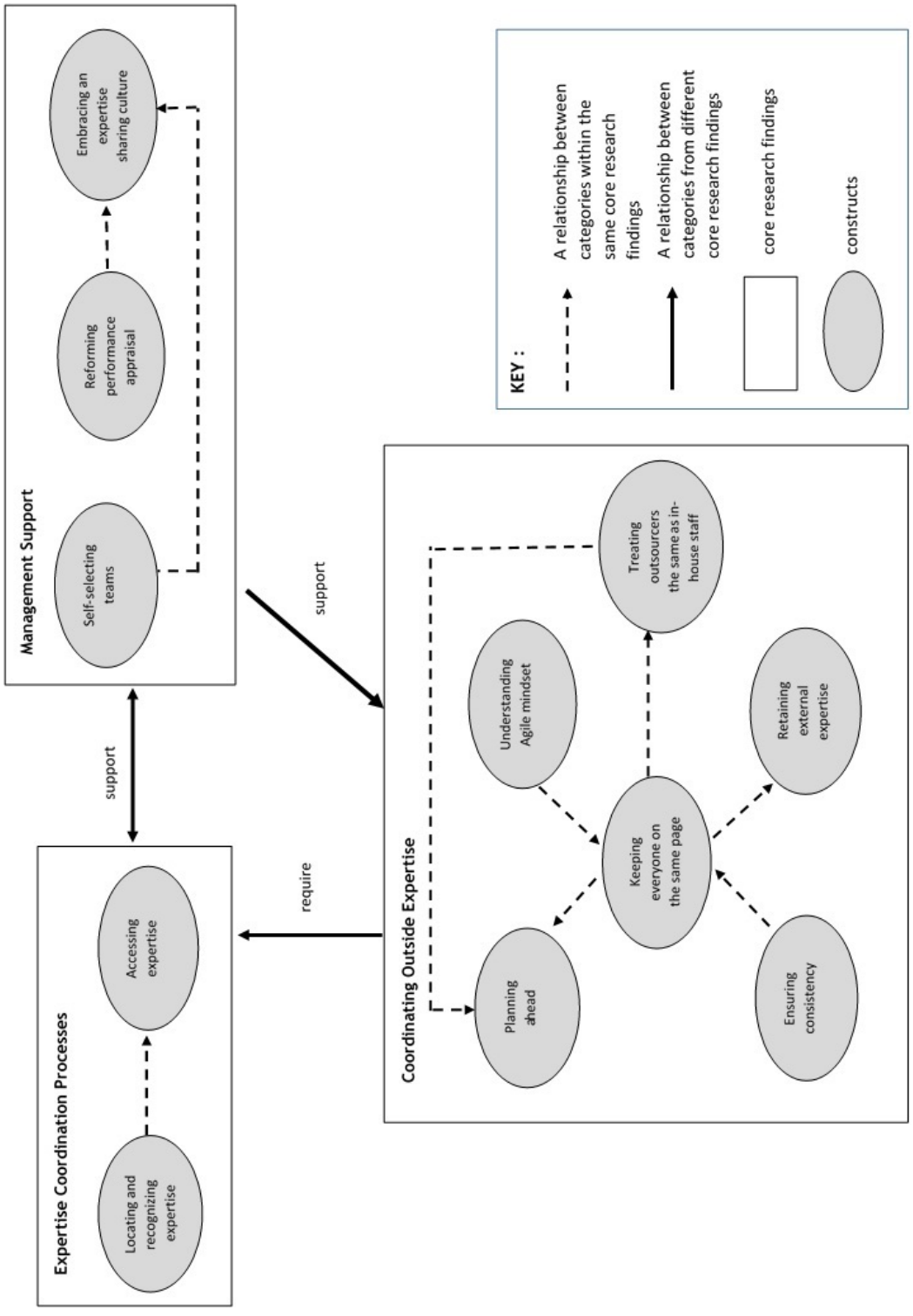


Figure 4.1: The Emergent Theoretical Code of this Study.

The findings of this study revealed two sequential steps in coordinating expertise:

- Locating and recognizing expertise
- Accessing expertise

According to Glaser's theoretical coding family [66], we identified that the *process family* fitted *locating and recognizing expertise* and *accessing expertise*. A sequential relationship exists between both steps. Determining the location of expertise by identifying the owner of expertise, and recognizing the need for it will assist in retrieving that expertise.

Locating expertise involves identifying who has which sort of expertise, and what is the level of that expertise, whereas recognizing expertise involves acknowledging the need for that expertise. There is no point having the expertise if teams are unable to acknowledge the importance of that expertise.

This study revealed six ways to identify and recognize available expertise in Agile teams as follows:

- Communicating frequently
- Working closely together
- Observing expertise
- Declaring self-identified expertise
- Earning certification
- Using an expertise directory

Locating and recognizing expertise assists in the next step of coordinating expertise, which is accessing expertise. Accessing expertise involves retrieving the existing expertise and pulling the new expertise into Agile teams. The findings of this study revealed five main approaches to accessing expertise as follows:

- Embracing expert-novice relationships
- Engaging hands-on learning
- Running effective meetings
- Establishing discussion channels
- Archiving explicit knowledge

4.3 Coordinating Outside Expertise

Expertise coordination must consider both internal expertise in teams, as well as external expertise which is located outside Agile teams. Besides in-house expertise, some companies use outsourcing as a solution to fill in the skills gap or staff shortages. Coordinating expertise outside Agile teams involves two different types of external expertise: external specialists and outsourcers. An **'external specialist'** is someone from outside the Agile teams but within the same organization, who supports Agile teams by bringing specialized skills such as user experience designers, database administrators, and software architects. An **'outsourcer'** is an individual or a group of people from external companies, who provide their expertise in Agile teams for a certain period of time depending on contract agreements.

This study describes how Agile teams, external specialists and outsourcers depend on each other to manage and utilize expertise outside teams. There are six strategies implied for coordinating outside expertise: *planning ahead (S1)*, *understanding Agile mindset (S2)*, *ensuring consistency (S3)*, *retaining external expertise (S4)*, *treating outsourcers the same as in-house staff (S5)*, and *keeping everyone on the same page (S6)*.

According to Glaser's theoretical coding family [66], we identified that the *strategy family* fitted *planning ahead (S1)*, *understanding Agile mindset (S2)*, *ensuring consistency (S3)*, *retaining external expertise (S4)*, *treating outsourcers the same as in-house staff (S5)*, and *keeping everyone on the same page (S6)*.

The relationships between the strategies of coordinating outside expertise are summarized in Figure 4.2. The oval shapes represent strategies, and arrows represent relationships between strategies. The figure depicts strategies involved in coordinating external specialists and outsourcers' expertise, which are relevant for coordinating both external specialists and outsourcers' expertise. A strategy of *treating outsourcers the same as in-house staff (S5)* is specifically applicable for coordinating outsourcers' expertise in Agile teams. Each strategy and their relationships are described in detail in Chapter 6.

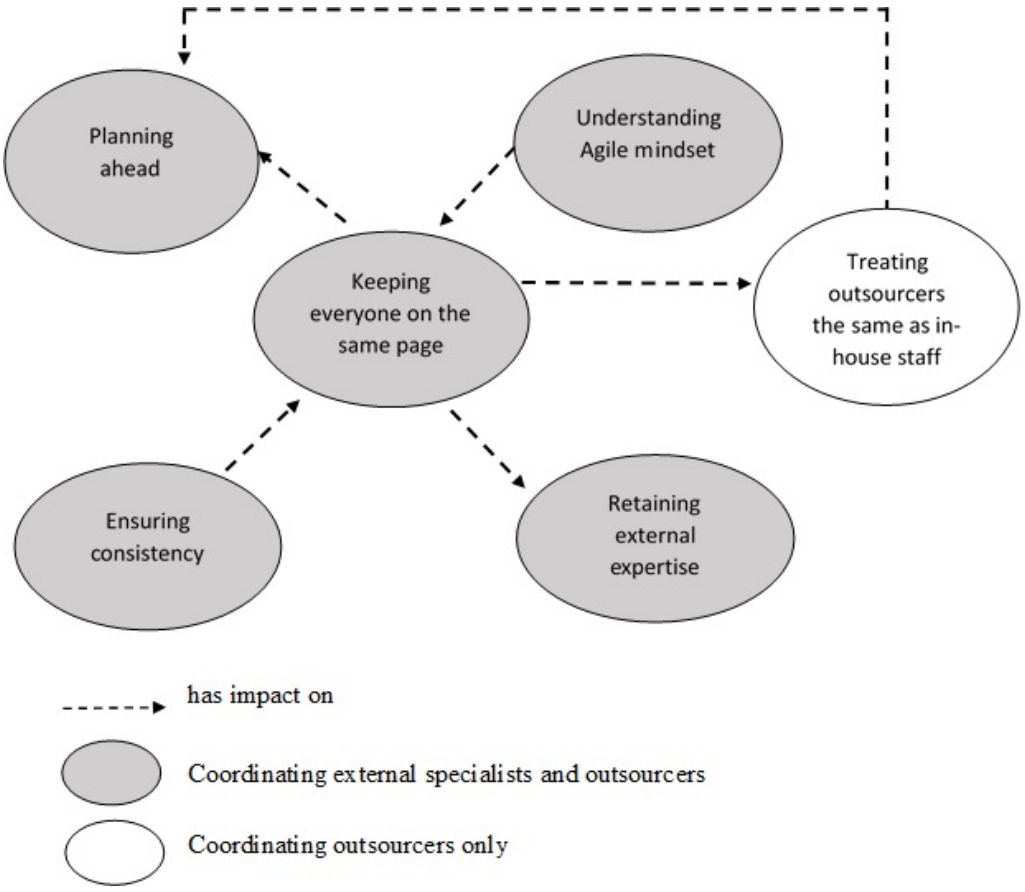


Figure 4.2: The Relationships between Strategies of *Coordinating Outside Expertise*.

4.4 Management Support

Successful expertise coordination relies on management support. There are three strategies of management support that emerged from the data analysis in supporting expertise coordination:

- Self-selecting teams
- Reforming performance appraisal
- Embracing expertise sharing culture

Management should promote self-selecting teams, which enable Agile team members to form their own teams. The right team with the right team members can be formed through self-selection, which assists Agile team members in sharing and accessing expertise when it is needed. The findings of this study revealed three activities required to facilitate self-selecting teams for Agile Software Development projects, which have a positive impact on expertise coordination as follows:

- Setting up cross-functional teams
- Selecting the right team
- Balancing teams

Through performance appraisal, management can support expertise coordination for Agile teams. Performance appraisal enables the development of individual expertise and the team's skills matrix. As Agile Software Development is team-oriented, Agile team members need to rely on each other in developing their expertise and team knowledge. This is the focal point where expertise coordination is needed in Agile teams. In order to implement an effective performance appraisal for Agile teams, the findings of this study indicated the need for reforming performance appraisal as follows:

- Integrating individual and team performance assessment criteria
- Shifting from quantitative to qualitative performance appraisal

An expertise sharing culture is vital in coordinating expertise for Agile teams, which requires team members to rely on each other. Management plays an important role in embracing an expertise sharing culture in Agile teams as follows:

- Fostering a supportive working environment
- Synchronizing goals

Based on Glaser's theoretical coding family [66], we identified that the *strategy family* fitted *self-selecting teams, reforming performance appraisal* and *embracing an expertise sharing culture*. These categories work as strategies for supporting *expertise coordination process* and *coordinating outside expertise*. The categories of *self-selecting teams* and *reforming performance appraisal* are related to *embracing an expertise sharing culture* and discussed thoroughly in Chapter 7.

4.5 Relationships Between Categories

Figure 9.1.1.4 depicts the relationships between categories, which are within and between the core research findings. Three basic relationships are involved between the core research findings:

- A bidirectional relationship between *expertise coordination process* and *management support*
- A unidirectional relationship between *management support* and *coordinating outside expertise*
- A unidirectional relationship between *expertise coordination process* and *coordinating outside expertise*

Successful *expertise coordination process* relies on *management support*. *Locating and recognizing expertise* is commonly associated with management in selecting the right people for the right software project. Turning individual knowledge into team-level knowledge through *accessing expertise* should be reinforced by management as well.

Management plays an important role in supporting the successful expertise coordination through *self-selecting teams*, *reforming performance appraisal*, and *embracing an expertise sharing culture.* The findings of this study indicate several features of *self-selecting teams* that foster the *expertise coordination process*. *Reforming performance appraisal* also contributes to the successful *expertise coordination process* through knowledge transfer criteria and qualitative feedback. On the other hand, *embracing an expertise sharing culture* supports the *expertise coordination process* through open work spaces, good personality traits, mutual goals, and team recognition.

The relationship between *expertise coordination process* and *management support* is two-way, in which the expertise coordination process also indirectly supports the implementation of *self-selecting teams* and *reforming performance appraisal*. Effective *self-selecting teams* and *reforming performance appraisal* rely on successful *locating and recognizing expertise*.

Besides the *expertise coordination process*, *management support* also tends to influence the success of *coordinating outside expertise*. A relationship between *coordinating outside expertise* and *management support* exists through strategies of *ensuring consistency* and *keeping everyone on the same page*. A *self-selecting team* tends to support the implementation of *ensuring consistency* strategy, whereas *embracing an expertise sharing culture* tends to support the strategy of *keeping everyone on the same page*.

There is a one-way relationship between *coordinating outside expertise* and *expertise coordination process.* Interactions and collaborations between Agile teams and outside expertise requires *locating and recognizing expertise.* Before engaging with tasks, both sides need to clearly define who they deal with, including roles and expertise. Similarly to *accessing expertise*, Agile

teams put emphasis on sustaining outside expertise by applying accessing expertise approaches in order to avoid too many dependencies on outside expertise. A detailed explanation on these relationships is described in Chapter 8.

4.6 The Theory Boundaries

As previously mentioned in section 3.4.4, the categories and their relationships define the boundaries of theory. The boundaries of this theory were determined at the beginning of this study; however, they were refined in order to align with the findings of this study. The boundaries of this study are twofold:

- Agile Software Development - the theory of this study emerged by considering the Agile values, practices, and principles. There is no restriction as to which Agile methods have been used, however, our findings were based on a broad range of Agile roles including expertise outside Agile teams.
- Expertise coordination - the theory of this study focuses on the specific coordination component, which is expertise. The findings of this study revealed that expertise coordination in Agile teams are involved in different layers of relationships, as summarized in Table 4.1.

Figure 4.3 illustrates the different layers of relationship that are involved when coordinating expertise in Agile Software Development projects. This illustration is not based on the actual findings of this study; rather it represents the level of coordination relationships we found in this study. Coordinating expertise begins in each Agile team by implementing the *expertise coordination process* for managing and utilizing team expertise. Depending on the number of teams existing in Agile Software Development projects, our findings indicated coordinating expertise also exists on an

Table 4.1: Relationship Levels of Expertise Coordination in Agile Teams.

Relationship levels	**Description**
Agile team members	an interaction between team members with common roles such as developers, software testers, and product owners for managing and utilizing team knowledge
Inter-Agile teams	an interaction between teams in an Agile Software Development project for managing and synchronizing team knowledge
Agile teams and outside expertise	an interaction between Agile teams and external specialists such as UX Designers and DBAs, as well as outsourcers

inter-team basis. Expertise coordination between Agile teams also requires *expertise coordination process* but with a slightly different purpose, which is to synchronize expertise between Agile teams.

Besides Agile team members and inter-Agile teams, expertise coordination also occurs when Agile teams deal with the expertise of outside teams. At this point, *coordinating outside expertise* emerged as a prevalent strategy for coordinating expertise outside Agile teams, as many issues arise when Agile teams and external specialists or outsourcers rely on each other.

Ultimately, in order to ensure successful expertise coordination, *management support* is essential to support *expertise coordination process* and *coordinating outside expertise*. *Management support* plays roles and responsibilities at each layer of the relationship in coordinating expertise. Chapters 5 to 7 give a detailed explanation of each category.

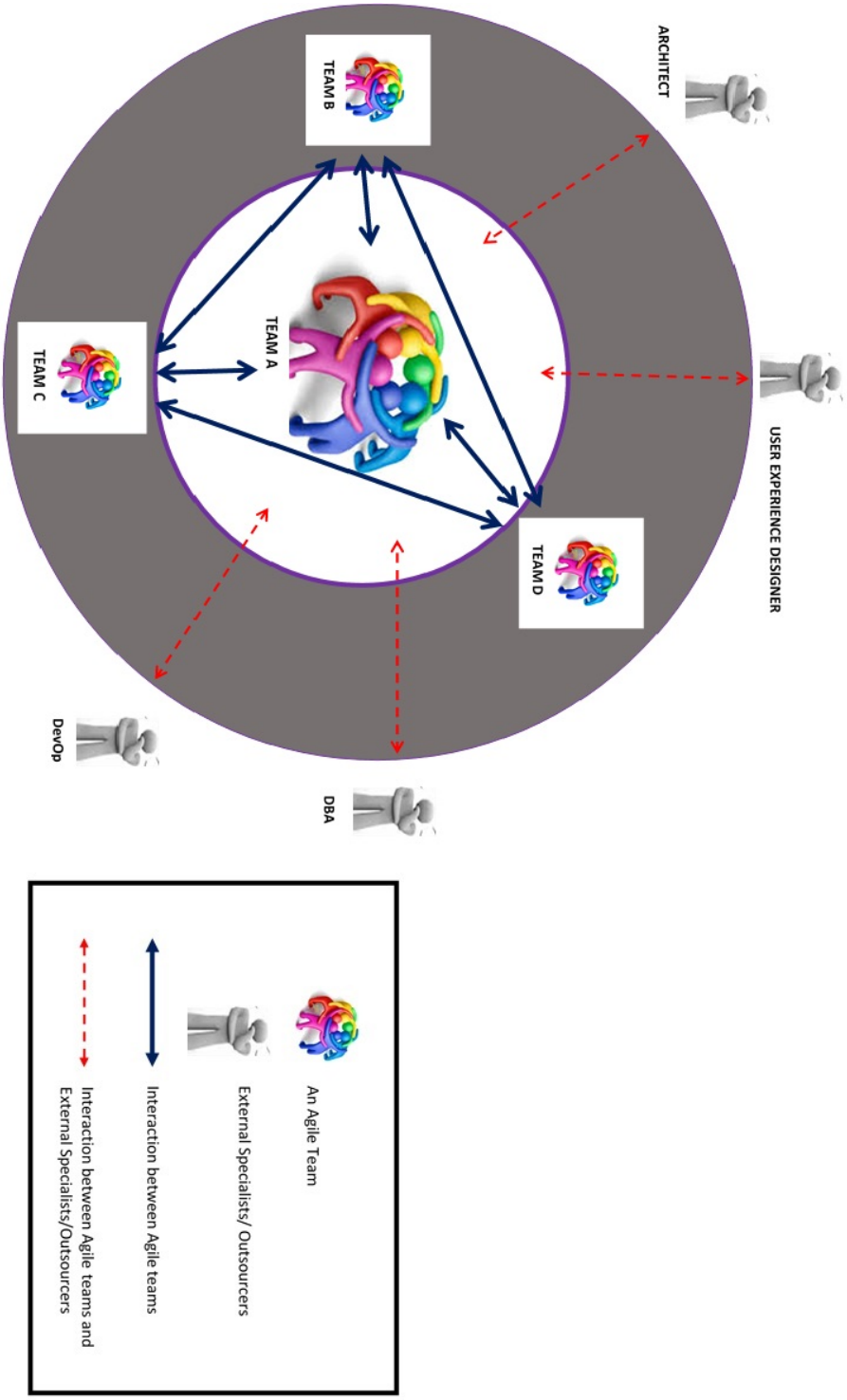


Figure 4.3: The Illustration of the Relationship Levels in Coordinating Expertise.

5

Expertise Coordination Process

This chapter presents the expertise coordination process for Agile Software Development Projects. As depicted in Figure 5.1, we found two steps involved in coordinating expertise:

- Locating and recognizing expertise
- Accessing expertise

The next subsections describe each step involved in coordinating expertise for Agile Software Development Projects, as well as the relationship between both steps.

5.1 Locating and Recognizing Expertise

In the context of this study, locating expertise refers to knowing who has knowledge and skills in Agile teams. Locating expertise is related to identifying who has which sort of expertise, and what is the level of that expertise. Locating expertise is commonly associated with management in identifying the right people for a new software project:

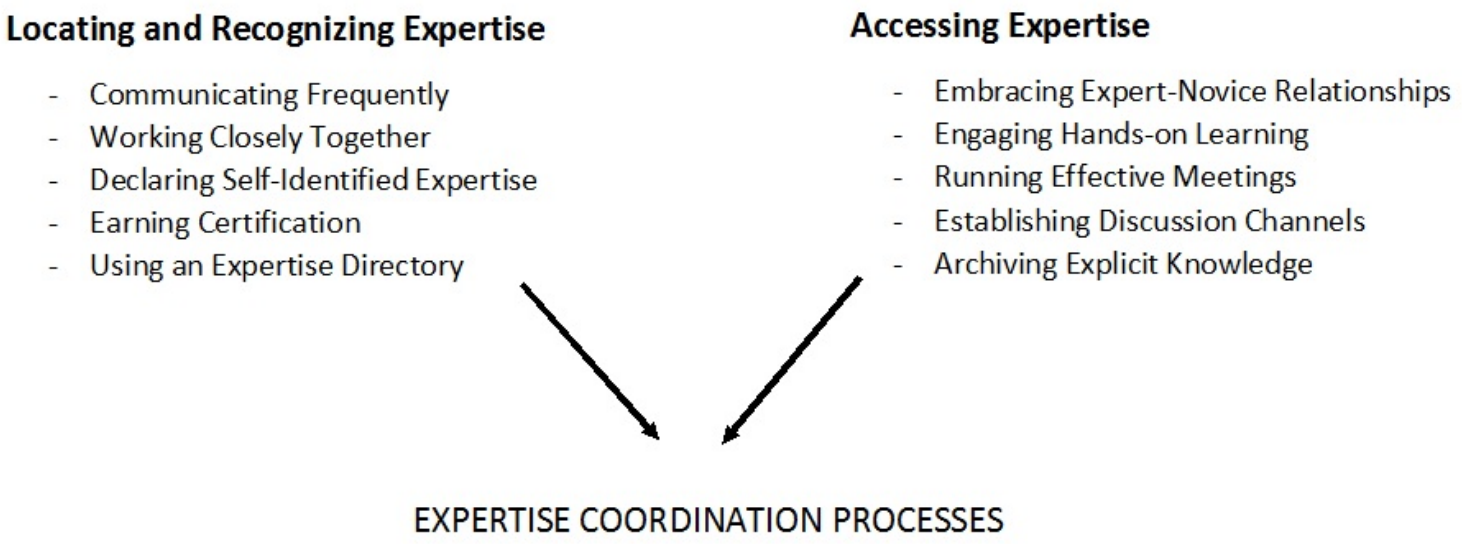


Figure 5.1: The Expertise Coordination Process for Agile Software Development Projects.

> *"That can happen in the project start up. When we start the project, we know that we need a certain skill set. Each team has a capability leader, who knows which level of expertise each of us has."* - P24, Scrum Master.

> *"Identifying expertise doesn't really fall on my shoulders. It is mainly for the project manager when he wants to bring the team together. He has to understand what the skills are and where the skills are needed for the project."* - P19, Business Analyst.

Knowing the location of expertise is not enough to coordinate expertise for Agile teams. In order to ensure the reliability of knowledge, expertise coordination requires Agile teams to recognize the available expertise. Recognizing and labelling an individual as an expert is often based on the degree of expertise, so, quantifying the level of expertise is essential in recognizing expertise.

We found that locating and recognizing expertise is beneficial for Agile teams to solve problems, select tasks, and develop expertise. Therefore, it is important for Agile team members to be aware of the development of their team members' expertise, in identifying and acknowledging relevant expertise.

The term *locating and recognizing expertise* emerged from the data analysis to describe how Agile team members locate and acknowledge the relevant expertise in Agile teams. This study revealed six ways to identify and acknowledge expertise in Agile teams: *communicating frequently*, *working closely together*, *declaring self-identified expertise*, *earning certification*, and *using an expertise directory*. Figure 5.2 illustrates the emergence of the category *locating and recognizing expertise* from the underlying concepts. These concepts are explained in the next subsections.

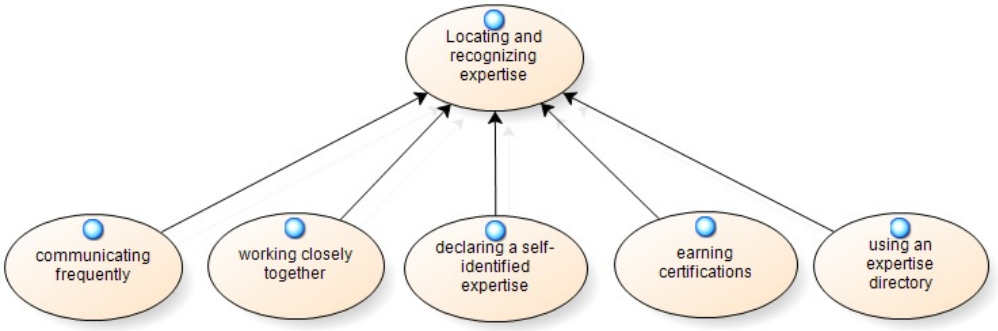


Figure 5.2: The Emergence of the Category *Locating and Recognizing Expertise* from the Underlying Concepts.

5.1.1 Communicating Frequently

A number of participants (eg. P2, P4, P6, P7, P8, P11, P21, and P24) noted that through frequent and effective communication, they could determine who in a team possesses particular knowledge and skills:

> *"Depending on what you need. Is it domain expertise? Is it expertise tool [sic]? Is it expertise with leadership and communication? You need to mix all [expertise] to be successful. So, talk to people and you'll find out"*- P6, Team Leader.

Agile team members can identify their team members' expertise by enquiring about the team members' backgrounds, including work experience, educational background, and proficient skills:

> *"In identifying the expertise [sic], just talk to people and ask 'what do you like to do?' or 'what are things that you have learn?'"*. - P7, Agile Consultant.

In certain circumstances, Agile team members can identify experts in a particular area based on recommendations of team members. Communication among team members provides a space to disseminate the meta-knowledge of the available expertise in Agile teams:

> *"You have people around, talking to you, and talk to that people, 'which an area of code is the best known to whom?' [sic]"* - P2, Agile Coach.

A daily stand-up meeting provides a communication vehicle for Agile team members to raise issues and obstacles that impede their progress. As time is limited, the focus should be on the identification of the right person with the necessary expertise and solution. This situation leads to divulging the available expertise in Agile teams:

> *"A daily stand-up meeting is not just reporting progress but [it is purposely] for team coordination. The main activity is to coordinate your work with the members of the team. For example, 'I have started with the [user] story but I'm stuck. Can someone help me?' "* - P4, Agile Coach.

The participants of this study, P11 and P24, affirmed that they obtained a second opinion from others to confirm someone's expertise, by referring to team members of previous projects, or getting some relevant information from management:

> *"In order to quantify their level of expertise, I just contacted the people who they have worked with."* - P11, Project Manager.

> *"That can happen in the project start up. When we start any project, we know that we need a certain skill set. Each team has a capability leader, who knows their level of expertise."* - P24, Scrum Master.

The information flowing freely through communication leads Agile team members to get to know who has which sort of expertise, as well as to recognize the available expertise. Through working closely together, the identified expertise can be confirmed and clearly quantified by peers.

5.1.2 Working Closely Together

Working closely together provides opportunities for Agile team members to identify and confirm their peers' expertise when collaborating together:

> *"We can actually work together and then doing programming together. We [work] in pairs, see and notice [the expertise]."*- P2, Agile Coach.

> *"Whether they really have the expertise or not, that's something that we can figure out day by day. If we work closely with them, day by day, we can clearly figure out their skill set."* - P36, Software Developer.

Through working together, the participants of this study (eg. P2, P4, P6, P14, P18, and P29) indicated that they could observe how their team members were working and delivering work outcomes. Therefore, Agile team members can acquaint themselves with the progress of expertise development of their peers. Collaboration provides a space to assess and quantify the team members' degree of expertise:

> *"I know that person best and what their level of expertise is when I see how they are working."* - P29, Software Developer.

For instance, one of the participants, P4, pointed out that they had the ability to identify and quantify their peers' expertise during sizing tasks, as well as selecting tasks:

> *"The expertise comes in discussion of the [story] size. If someone is less experienced, they might put the size differently from the experienced person."* - P4, Agile Coach.

> *"....when the stories come out, they choose stories or works based on their capability."* - P4, Agile Coach.

This is aligned with the observation findings, which show that team members were able to determine the capability of their team members during sizing tasks:

> *During planning poker, the developers who had selected a different value of estimation, needed to discuss their estimation. The high and low estimators explained the reason of their estimation. Most of the time, they related technical skills feasibility to the complexity of user story in the discussion.* [Observation notes]

In order to confirm and assess their peers' expertise, participants indicated that they used demonstrations to ensure the competence and credibility of their peers:

> *"I trust what they know about what they are doing. So, I asked them to demonstrate [their work] to other people. So it is a part of process for me to understand what they have done, and for me to present [their work] to other teams or customers. That's typically enough to make sure the scope that we discussed is happening."* - P14, Product Owner.

Most Agile activities and practices encourage Agile team members to work together. There is no doubt that identifying expertise areas and levels can be obtained through effective collaboration in Agile teams.

5.1.3 Declaring Self-Identified Expertise

A curriculum vitae is used to represent the expertise details for recruitment purposes. In order to succeed in an interview, Agile team members convincingly disclose their self-identified expertise to interviewers. This process provides a clear picture of Agile team members' expertise at the very beginning:

> *"They are telling you what they are good at. For example, 'I'm a great .NET developer'. So, start with that [declaration], put them into the role, and observe them and see what they can do."* - P4, Agile Coach.

After joining an Agile team, Agile team members tend to expand their expertise into other expertise areas. Some team members, however, are not aware of the development of their team members' expertise. This is the point where Agile team members need to declare their expertise, in order to let others know what they can contribute to Agile teams:

> *"We do a stand up meeting...we talk about challenges that we have faced. Someone might know what the issue is about. They just said 'Yup, I know about that. We can talk about it later.' "* - P5, Software Tester.

The main concern is how reliable the self-declaration expertise is, as it is based on individual judgement. Therefore, it is important to build mutual trust of self-declared expertise. Mutual trust is a shared belief that Agile teams can rely on their peers' expertise. Pretending to be an expert does not allow trust to flourish in Agile teams. A few participants of this study (eg. P1 and P18) claimed the existence of this situation in their teams, which engendered a lack of trust by disseminating incorrect knowledge and skills in Agile teams:

> *"If someone asks a question, and we are not sure what the answer is, we do not pretend like we know the answer."* - P1, Software Developer.

> *"I found some people who are talking like they know the programming language, but they actually don't understand."* - P18, Software Developer.

Building trust is not simple and sometimes takes time to develop. The same scenario occurs when acknowledging self-declared expertise. In certain circumstances, Agile teams need the space and time to prove their skills and competencies:

> *"That's what I do to bring out people's skills. You have to watch, see what works and what doesn't. If something doesn't work, don't*

immediately blame them. Try a different approach. You have to take on trust what they tell you what they are good at." - P4, Agile Coach.

Mutual trust is hard to gain unless Agile team members are willing to be open and honest about their expertise. In order to ensure the reliability of self-identified expertise, it is essential to verify the expertise through communicating frequently and working closely together, as well as earning certifications.

5.1.4 Earning Certifications

Earning software development certifications is another indicator that leads Agile team members to identify their team members' expertise. Someone who earns a technical certification shows the ability to master a new body of software development knowledge:

"It is good to have a certification especially for personal development, and when we have the certification, it shows that we have the knowledge." - P33, Project Manager.

Through communication and collaboration with others, Agile team members can learn from those with certifications in a particular area of expertise. Moreover, management effort in informing Agile teams about the status of team members in obtaining certifications tends to assist team members in locating expertise:

"The human resources manager will announce that this person will go to study and get a certification." - P33, Project Manager.

Earning a certification normally involves costs which require expenditure on study materials, training costs, and testing fees. Hence, one of the participants, P33 argued that not everyone has the opportunity to be certified and an organization has to give priority to someone who needs the

certification. Beside cost, time is another factor that hinders someone from obtaining a certification. As there are often heavy workloads and deadline pressure, some Agile team members were not able to get a certification:

> *"But it is not certainly true. The certification is costly and it might be there is not enough time to get the certification."* - P33, Project Manager.

Therefore, obtaining a certification is not a reliable measurement of expertise. Without a certification, someone can still perform well in his or her field of expertise. Earning certification, however, is an accreditation for people to recognize expertise.

> *"I learn a lot through Scrum training and certification. But if these become requirements, that's against my principle. Everyone can learn. In fact, in my team, there are two staff who do not have computer backgrounds, but they can [still] do their job. Without a certification, they still can do the job."* - P32, Scrum Master.

Hence, earning certification should be aligned with experience gained. Participant P31 claimed that he practised what he has learned and tried to connect with others in exchanging knowledge and experience after obtaining certifications:

> *"I found that getting a certification is helpful because it is structured but, to really understand the practices, we need to talk to other people. Everybody has to try it and learn it in a different context. What I did after getting the certification is talking to other practitioners. What they have tried and what I have tried, and what does work and what doesn't work."* - P31, Scrum Master.

Even though the findings of this study indicated different perceptions of earning certifications, there is no doubt that Agile team members can locate

expertise based on certifications earned. Certifications measure Agile team members' knowledge and skills against software industry benchmarks to prove they have the right expertise. The reliability of the certifications, however, relies on how the individual utilizes and practices the knowledge gained in a software development environment.

5.1.5 Using an Expertise Directory

Our findings revealed the usage of an expertise directory to identify where expertise resides in Agile Software Development projects. An expertise directory consists of expertise profiles of Agile team members. We found that management preferred to have an expertise directory when selecting the right person with the right expertise for upcoming software projects:

> *"We do have the skills database. We developed [it] ourselves. Each person is expected to keep his or her profile and can be searched by others. If I want to start a new project and I need X, Y , Z skills, I guarantee that the skills database can provide these skills."* - P11, Project Manager.

An expertise directory also provides valuable information for management in determining the lack of expertise in Agile teams. This will assist management to take further steps in bridging the expertise through expertise development such as training, mentoring, and coaching:

> *"If they can't find enough people for the project based on the skill set, then they send [someone] for training. It could be a person who has rated himself '1' for that skill. Then, the company should invest more in him, because he has some knowledge but he is not an expert. Then, the company send him for training [sic]."* - P33, Project Manager.

Besides management, Agile team members can also seek the available expertise information through the expertise directory for getting assistance from an internal expert:

> *"Expertise directory is good for me. For instance, I don't know Python. So, I can find people who know Python through the expertise tool."* - P18, Developer.

Agile team members are required to update their expertise profiles to enable them to be selected for upcoming projects when their expertise is aligned with the project requirements:

> *"They are motivated to update their skills database. So, that means, for the next exciting project, if they have the skills, they have a chance to be called to join the project."* - P11, Project Manager.

Therefore, management should play an important role in setting up proper workflows of updating the expertise directory. In order to ensure the involvement of every Agile team member in updating the expertise profile, a compulsory guideline is applied in implementing the expertise directory:

> *"We have a set of questions, which is a sort of skill set. We have to do that every year. Let's say, I am good in .NET. So, I have to rate myself, between 1 to 5. At the end, here is the result. Then, it goes to our manager. The manager will compile everything."* - P31, Scrum Master.

> *"There is no issue in using the skill set because it is compulsory to use. Everyone has to submit eventually."* - P31, Scrum Master.

Even though an expertise directory is beneficial to Agile teams, several participants claimed that communication is more valuable when finding relevant expertise than using an expertise directory. This finding reveals that relying on the expertise directory to find the relevant expertise is not the best option in Agile teams:

> *"You need to talk. It is much more effective than to go to some expertise repository. How do you know the 'cruft' factor of the expertise repository [sic]?"* - P2, Agile Coach.

Several participants (eg. P2, P16, P17, P18, and P39) indicated that there was a formidable challenge in using the expertise directory to search for the relevant expertise in Agile teams. The main issue was the reliability of the expertise directory in providing accurate expertise profiles. The expertise directory required regular maintenance to update the expertise profiles. The updating task needed someone in the Agile team to act as an administrator to maintain the tool and this indirectly increased the workloads of the Agile teams:

> *"For example, I'm a beginner in Java programming. Then after three months, I continued improving Java programming. So, who is going to update the repository?"* - P2, Agile Coach.

> *"There is a problem with updating expertise details. Just once a year based on a reminder from the company, which comes through email. It is quite time consuming to maintain it."* - P18, Developer.

The organization's preference in using the expertise directory depends on the size of the organization. The growth of an organization with an increased number of staff and projects contributes to the high possibility of using the expertise directory:

> *"We realized that we have many projects coming. So, it is important to get alignment between the project and people. When we started, we had 110 people, and now 200. It becomes harder to keep track of a lot of skills. So, this tool helps me to identify the skills."* - P11, Project Manager.

Furthermore, implementing an expertise directory, either manual or computerized, depends on the needs of and benefits to the organization and Agile teams:

> *"When it is started, it was an excel application. Then, the application was moved to the SharePoint. This company is big and involves many different locations, and very much distributed projects [sic]."* - P33, Project Manager.

Access to internal experts through an expertise directory tends to speed up expertise searching. The necessity of having an expertise directory, however, relies on the organization's need to locate a source of their staff's expertise.

5.2 Accessing Expertise

Diversity of expertise leads to the distribution of expertise in Agile teams. Expertise distribution requires dependencies among team members in accessing expertise. Most Agile team members (eg. P1, P2, P4, P31, P33, P38, and P42) indicated a positive attitude to accessing expertise.

> *"There is no issue in sharing knowledge because we help each other. If they don't help me today, I won't help them tomorrow. We help each other because it is beneficial for both of us. If they fail, I will fail."* - P31, Scrum Master.

> *"I didn't have a situation where my team members didn't want to share their knowledge with me."* - P1, Software Developer.

The purpose of accessing expertise is to support dependencies among team members in retrieving the available expertise, as well as new expertise. Lack of expertise requires pulling new knowledge and skills into Agile teams. Accessing expertise involves retrieving the existing expertise and pulling the new expertise into Agile teams.

The category *"accessing expertise"* emerged from the data analysis to describe how Agile team members disseminate available expertise and

pull new expertise into the team. Even though there are many ways of accessing expertise in Agile teams, the findings of this study revealed five main approaches to accessing expertise: *embracing expert-novice relationships*, *engaging hands-on learning*, *running effective meetings*, *establishing discussion channels*, and *archiving explicit knowledge*. Figure 5.3 depicts the emergence of the category *"accessing expertise"* from the underlying concepts. The concepts are described in the next subsections.

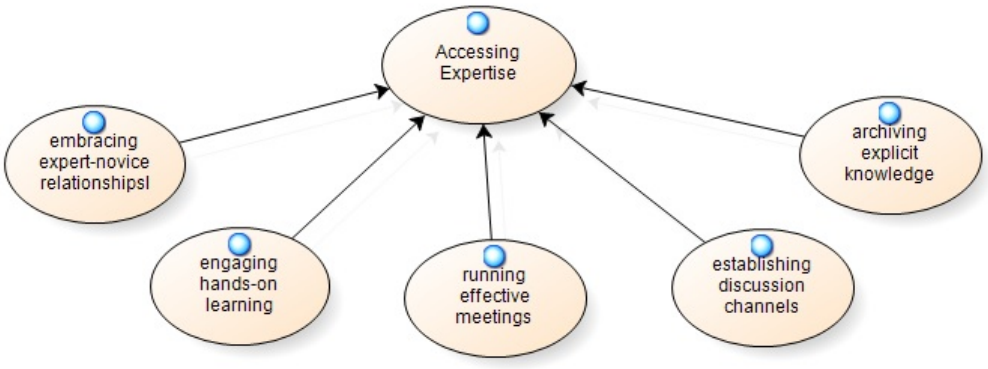


Figure 5.3: The Emergence of the Category *Accessing Expertise* from the Underlying Concepts.

5.2.1 Embracing Expert-Novice Relationships

An expert-novice relationship allows accessing expertise in Agile teams by disseminating existing expertise and pulling new expertise into the team. The findings of this study revealed three basic relationships between an expert and novice for Agile teams: *apprenticeship*, *mentorship*, and *coaching*.

5.2.1.1 Apprenticeship

A few participants (eg. P3 and P39) reported that master and apprentice relationships focus on pulling new expertise into Agile teams. A master

is an internal or external expert who has the responsibility to bring new expertise into Agile teams, whereas the apprentice is someone seeking new expertise. Learning by working together is the focal element in the apprenticeship:

> *"The master is where you pull skills you don't have [sic]. Someone internal, or you, might bring contractors or a consultant for a small period of time. They start work together. The apprentice learns as well as he can and learns to become a master."* - P3, Agile Consultant.

The master-apprentice relationship is a temporary scenario, whereby the master will leave the Agile team once the apprentice has grasped the new expertise. In order to disseminate the expertise widely into Agile teams, the same implementation can be repeated by rotating the apprenticeship with other team members. At this point, the apprentice becomes a master and trains others based on what he or she has learnt:

> *"When the expert goes, the team [still] has knowledge of how to do the piece of work."* - P3, Agile Consultant.

A different perspective was pointed out by participant P39, regarding the relationship between master and apprentice. Even though a master-apprentice model emphasizes learning by collaborating, the relationship between master and apprentice is basically an authoritarian one-way relationship. The master is dominant and has full control of the learning process, whilst the apprentice has to grasp as much knowledge as he or she can depending on his or her ability:

> *"The only concern that I have about the master-apprentice program is a hierarchy-relationship, in which the master is a master and where the apprentice is a student. It looks like 'I'm the person who knows how to do that and you're not. So, you have to learn'."* - P39, Agile Coach.

Developing a learning organization is a benefit gained from the master-apprentice relationship. A learning organization creates a culture that encourages and facilitates continuous employee learning. Through a learning organization, Agile team members tend to transfer from individual learning to shared learning.

5.2.1.2 Mentorship

Mentorship involves a relationship between mentor and mentees for personal and career development. A mentor is a more experienced and knowledgeable person, whereas a mentee is someone who needs guidance and assistance from the mentor in a particular area of expertise. Unlike a master-apprentice relationship, mentoring relationships are often unpaid and on a voluntary basis. Therefore, mentorship enables accessing expertise in Agile teams through an informal transmission of knowledge between mentor and mentee:

> *"The mentor-mentee relationship focuses more on the development of expertise and what they want to learn."* - P29, Developer.

Working on same role and area of expertise should be taken into account in matching mentor and mentee. This will provide an opportunity for the mentee in seeking the mentor's help and indirectly tends to strengthen the mentee's knowledge and skills:

> *"The mentoring is applied into a certain discipline and shares the discipline, because the goal of mentoring is focused on a discipline."* - P39, Agile Coach.

> *"Mentoring is usually in the same role."* - P46, Agile Coach.

The findings of this study, however, indicated that some mentees preferred to be matched with a mentor who had a different role and background of expertise. A different role and expertise lead to different

objectives for the mentorship, which enables the mentee to explore another area of expertise and develop a diversity of skills:

> *"In some exceptional circumstances, there is a good relationship between developer and Business Analyst. They just get on the right mix that they can immerse. Most of the time, it turns up to the same skills."* - P46, Agile Coach.

An effective mentorship can be achieved when mentees are allowed to select a mentor based on their own development needs and interests. The greater the involvement of the mentee in the selection of their mentor, the better the outcome of knowledge transfer between mentor and mentee:

> *"Normally, mentor must be a senior and mentee is a junior. It is up to the mentee to choose his mentor. In other way around, the mentor also can choose his mentee."* - P46, Agile Coach.

> *"The mentee chooses his or her mentor."* - P39, Agile Coach.

There is no formal way of implementing mentorship for Agile software projects. The biggest concern in mentorship, however, is how to ensure the mentee is matched with the right mentor, which tends to influence the successful expertise sharing between the mentor and mentee.

5.2.1.3 Coaching

Coaching requires experts' involvement in disseminating their expertise to Agile teams:

> *"If the team doesn't know anything about the skill, the best thing is to invite a coach from the real community and teach us [sic]."* - P2, Agile Coach.

Agile teams adopt coaching through Agile coach roles. An Agile coach plays an important role in facilitating team members to gain new knowledge and skills. An Agile coach can also act as a mentor in strengthening team members' skills, particularly for those who have failed to meet expectations. With the proper guidance and adequate expertise, team members can gain benefits through the coaching approach:

> *"Bring the coach for several times, and pair and teach [them] how to do [sic]."*- P6, Team Leader.

The findings of this study indicate that there are a few differences among apprenticeship, mentorship, and coaching. Apprenticeship and coaching focus more on pulling new expertise into Agile teams, while mentorship concentrates on disseminating existing expertise. While all approaches aim to disseminate expertise, the apprenticeship puts a high priority on producing an expert or a master in the specific skill.

In general, apprenticeship and mentorship both involve a relationship between two parties; however, there is a difference between both approaches. The relationship between master and apprentice is a hierarchical relationship, which is based on authority, whereas mentorship focuses on a personal development relationship. Apprenticeship and coaching relationships last as long as they are needed, depending on the purpose of both relationships. Successful mentoring, however, often lasts up to a year, enabling mentors and mentees to learn from one another.

In terms of implementation, mentorship and coaching focus on facilitating and training, whereas the apprenticeship emphasizes learning by collaborating. Apprenticeship and coaching require hiring external individuals to pull their expertise into teams, while mentoring concentrates on disseminating available expertise from internal individuals within the organization.

The role of a coach is to guide teams to continuously improve their Agile practices. Apprenticeship and mentoring, however, emphasize de-

veloping technical skills. Coaching focuses on the whole team, whereas apprenticeship and mentoring concentrate on individual.

Even though there are differences between apprenticeship, mentoring, and coaching, all approaches complement one another as each approach focuses on different purposes in disseminating expertise in Agile teams. Table 5.1 summarizes the differences between apprenticeship, mentoring, and coaching.

Table 5.1: The Differences between Apprenticeship, Mentoring, and Coaching.

Criteria	**Apprenticeship**	**Mentoring**	**Coaching**
Purpose	Pull new expertise	Disseminate available expertise	Disseminate available expertise
Relationship	Authority and hierarchy relationship	Personal development relationship	Task-oriented relationship
Duration	Short-term	Long-term	Short-term
Implementation	Learning-by-collaborating	Facilitating and training	Facilitating and training
Source of expertise	Internal and external expertise	Internal expertise	External expertise
Type of expertise	Technical skills	Technical skills	Agile practices
Targeted trainees	A particular individual	The whole team	A particular individual or team

5.2.2 Engaging Hands-on Learning

In order to strengthen learning through an expert-novice relationship, hands-on learning methods can be incorporated into apprenticeship, mentorship or coaching. Transforming the learning experience through hands-on exercises provides opportunity to develop expertise, as well as distribute expertise. Drawing from the research findings, there are four ways to engage hands-on learning in Agile teams: *pair-programming*, *coding dojos*, *design exercises*, and *internship programmes*.

Pair-Programming Most participants (eg. P1, P2, P3, P8, P11, P13, P18, P21, P26, P27, and P29) affirmed that pair-programming is a powerful practice in disseminating their expertise to other team members:

> *"Learning from one another through pair-programming. Keep pairing and everybody will learn and pull expertise."* - P2, Agile Coach.

Pair-programming enables Agile teams to share expertise and indirectly speeds up the work. We found evidence of the usefulness of pair-programming, as shown in our notes during observations at company XYZ:

> *During sprint planning, the product owners asked the developers to clarify the unfinished tasks before they proceed with the new user stories. One of the developers mentioned that there were too many user stories for the last sprint. It was hard for him to cope with a lot of tasks. Then, the product owner decided which tasks need to be finished in the current sprint by asking the developer to work in a pair with another developer*. [Observation notes]

Coding Dojos A coding dojo provides a space for Agile team members to learn, practise, and share their programming skills. Pair-programming is

embedded in coding dojo practices. A coding dojo involves two team members working together to solve the programming problem with feedback and guidance from an audience, mostly other developers. Role rotation is essential to enable audience members to have hands-on programming within the limited time:

> *"The developers run workshops, [by] running a coding dojo. Then, they will be paired on the computer to tackle some problems. We rotated the team around and we can see what other people are doing."* - P15, Agile Coach.

Design Exercises For designing software, several participants indicated that they used design exercises for capturing the customers' preferences in software design. Design exercises enable customers to put forward possible ideas for designing the software. Agile teams, particularly designers, can articulate the benefits of design exercises by understanding the customers' needs and sharing and exchanging their expertise with customers:

> *"We start with branding the logo, colour, and background. So we can feel what kinds of design and style that they want. We run a workshop for a few weeks. We do exercises with magazines and ask them to cut out things that they like. Those influence the needs of the software to be presented. Then, we let them explain and we get the idea from that."* - P20, User Experience Designer.

> *'We used the design exercise to get some very initial feedback from various parties. Then, it goes to the Scrum team and we also get feedback from them."* - P14, Product Owner.

Internship Programmes Hands-on learning can also be gained through an internship programme. This programme requires Agile team members to attach themselves to other teams for a short period of time. The interns

need to collaborate and gain experience and knowledge through a different working environment. The ultimate goal of the internship programme is to bring new knowledge and skills into Agile teams:

> *"Internships are like short apprenticeships where people can learn from people who are really good at something by working alongside them. They temporarily join another squad for two to four weeks and learn as much as they can."* - P12, Agile Coach.

> *"We do a lot of exchanges. For the last six months I have been working in Bangalore. When I came back I can let others know about what I have learnt. Learning more about other cultures. We learn a lot when we come from a different environment."* - P30, Business Analyst.

These hands-on learning methods can be incorporated in expert-novice relationships. For instance, pair-programming is used in implementing an apprenticeship, as an apprenticeship emphasizes learning by collaboration. However, most identified hands-on learning methods are applicable to foster shared learning even without master or coach involvement.

The findings of this study indicated four ways to engage hands-on learning in Agile teams: *coding dojos, pair-programming, design exercises* and *internship programmes.* There are probably other hands-on learning methods available for Agile teams which were not discovered by this study. The emergent hands-on learning methods offer another approach for accessing expertise in Agile teams.

5.2.3 Running Effective Meetings

Effective meetings allow for open conversation that draws upon Agile team members' knowledge and skills to solve problems arising during the development of software projects. Running effective meetings indirectly leads to accessing expertise among Agile team members.

The findings of this study indicated the importance of Agile meetings, such as a daily stand-up meeting, sprint planning, retrospective and sprint review. Most participants (eg. P5, P11, P15, P21, P22, P27, P39, and P41) indicated that these meetings allow for open conversation that draws upon Agile team members' knowledge and skills to solve problems arising during the development of software projects. Solving the problem often requires accessing expertise among team members. There was evidence of accessing expertise during Agile meetings, as shown in our notes on Team A during observations at company XYZ:

> *During the sprint review of Team A, one of the developers addressed a part of the current system, which required a clarification from the product owner. After listening to the product owner's explanation, he found that he might be unable to rewrite the code for that part of the system. Then, everyone in the meeting tried to provide suggestions and opinions for solving the issue. Most of the time, however, they relied more on the product owner to provide solutions on how to solve the issue arising including technical skills.* [Observation notes]

Observations and interview results consistently showed that Agile meetings enable team members to find solutions for issues or problems arising. Agile teams, however, should avoid predominantly relying on a single point of expertise for solving problems. This situation could prevent the richness of team knowledge, because the ideas were mainly from a single point of view. When Agile teams are overdependent on a single expert, there is a possibility of being unable to proceed to the next stage of tasks, particularly when facing problems or issues without the assistance of that person.

In contrast at Team B, the product owner addressed an issue related to unclear and ambiguous user stories during sprint planning (see Figure 5.4). He asked for opinions from developers to improve the user story. This is a good practice of accessing expertise, which involves all team

members contributing their knowledge. Every member takes responsibility for all user stories and puts effort in to understanding each user story, and indirectly supports collective code ownership. Our observation notes also indicate another incident that facilitated collective code ownership:

> *During the sprint planning, the developers of Team B asked the product owner for a clarification on sprint backlogs. When discussing each sprint backlog, we found all developers kept asking questions to deeply understand the backlog. This information was useful to assist them in listing and sizing tasks related to the sprint backlog (see Figure 5.5).* [Observation notes]

Based on the above observation notes, we found it was hard for us to determine who was responsible for a specific sprint backlog. The code was owned and shared by the entire team, and every team member had permission to change and refactor the code (refer to section 2.3.2). Accessing expertise involves inter-dependencies between team members in sharing their knowledge and indirectly facilitates collective code ownership.

The observations also revealed that Agile meetings were used to discuss how to access internal or external expertise, when there was insufficient capacity of expertise. Figure 5.6 shows the stand-up meeting of Team A and led to the following observation notes:

> *During the stand-up meeting, the product owner had been informed that one of the developers was unable to work for the current sprint. In order to deal with insufficient capacity of expertise, they agreed to access expertise located outside their team.* [Observation notes]

In order to manage teams involved in an Agile software project, Participants 41, indicated the importance of Scrum of Scrums meetings in coordinating tasks and expertise:

> *"They have a Scrum of Scrums meeting, which all Scrum masters [need to] attend the meeting twice a week. They talk about what the*

Figure 5.4: The Sprint Planning Meeting of Team A.

team blockers are. Maybe there are dependencies between blockers. We need to know what dependencies are related to." - P41, Agile Coach.

The Scrum of Scrums meetings enable all teams to coordinate tasks and identify impediments that block the progress development of the software project. The representatives of each team (such as the Scrum Masters) discuss possible solutions for managing the defined impediments. At this point, finding solutions requires sharing and integrating the expertise of the team representatives.

Besides Scrum of Scrums meetings, there are specific meetings for certain Agile roles such as developers' meetings and testers' meetings.

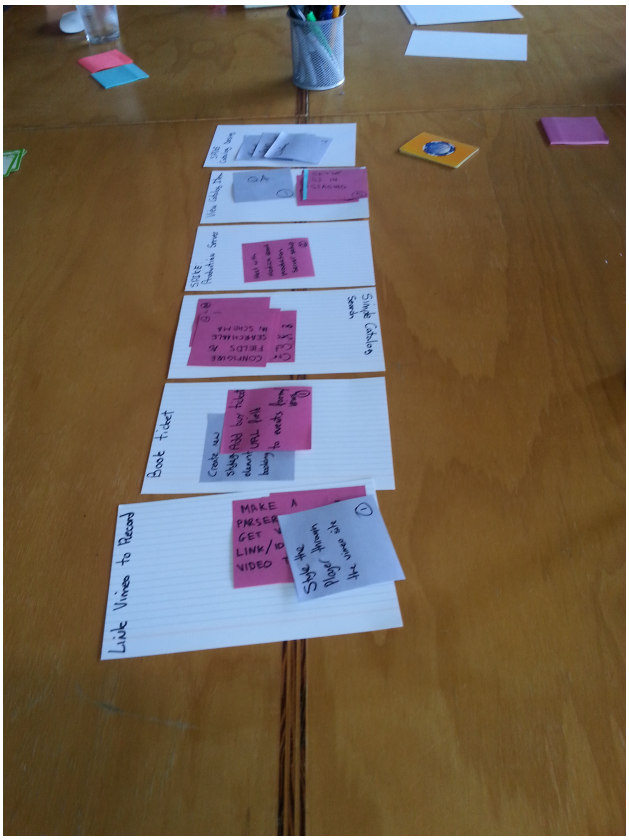


Figure 5.5: A List of Tasks for Each Sprint Backlog

Even though these meetings are more applicable for traditional software development projects, Agile software projects, particularly large-scale application-development projects, really need these meetings. While Scrum of Scrums meetings focus on teams in general, these meetings emphasize task coordination and problem solving in a specific Agile role:

> *"The testers have the testing meetings. The developers also have the development meetings. We do a round table to update stuff. If there is something causing a big problem, then, we try to discuss about that."*
> - P45, Software Tester.

Furthermore, specific Agile role meetings provide a space for learning

Figure 5.6: A Stand-up Meeting of Team A.

and sharing expertise among Agile team members within the same role:

> *"In order to share knowledge, to grow and to learn, all Business Analysts meet at least once a week."* - P46, Agile Coach.

For learning and sharing expertise, a demonstration session is integrated in intra-team meetings. A demonstration session is also useful for presenting the progress of tasks and indirectly assists all teams to determine the overlap and dependencies between tasks. Therefore, the dependencies between tasks and who is responsible for each task can be easily visualized as depicted in Figure 5.7:

"We actually have a weekly meeting and we share design critique. We show what we have been working on for the week. I found there is a tool that helps me a lot in this project. So I demo the tool to my team mates." - P38, User Experience Designer.

"Every week we have a demo session. For example, our team worked on search function. The other team came into the session and we showed them what we had. So, they could see what other teams were working on. They knew what will affect their tasks rather than the Scrum master talking to the team. Then, the entire team knew what was affecting the other teams. So, they knew who they can talk to." - P47, Developer.

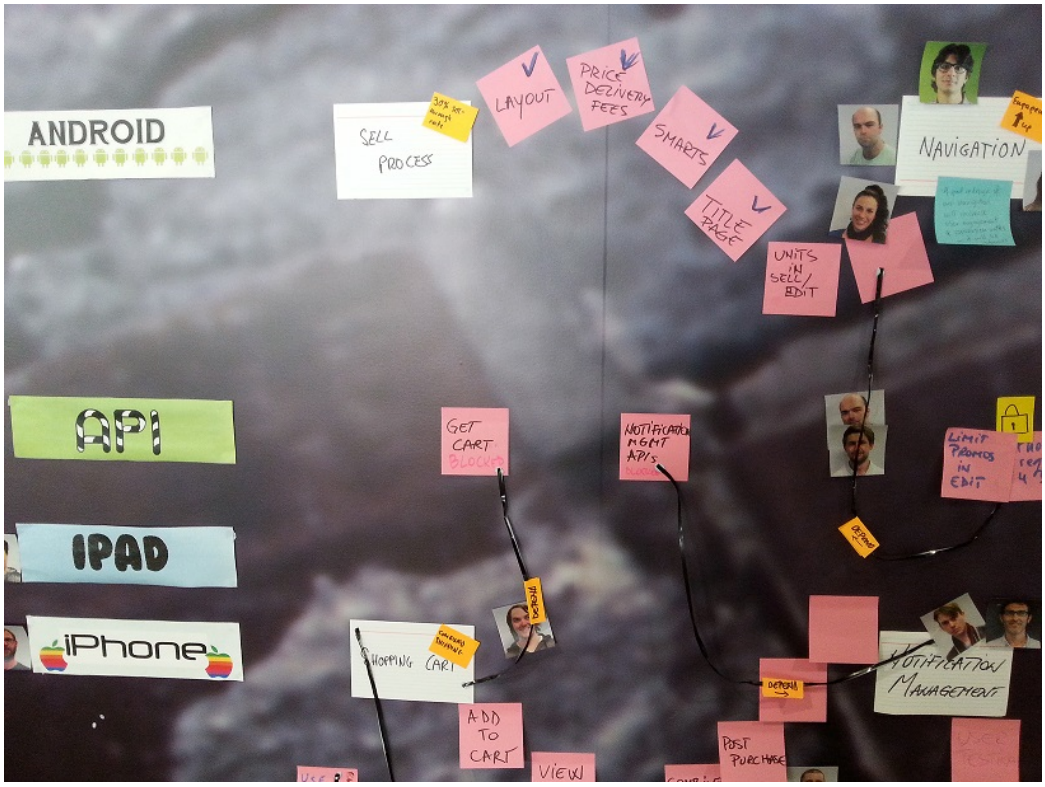


Figure 5.7: The Dependencies between Tasks.

Agile teams should understand the purpose of each meeting and figure out the best way to run the meeting effectively. Integrating rules and procedures tends to assist Agile teams to run effective meetings and indirectly improves accessing expertise through meetings.

5.2.4 Establishing Discussion Channels

Having meaningful discussions through the right platforms helps to enable distribution of expertise in Agile teams. Four discussion channels have been identified from this research finding: *interest groups*, *chat tools*, *knowledge sharing sessions*, and *robust debate.*

Interest Groups The findings of this study revealed that interest groups provide a solid base of interconnection among peers who have similar interests in a particular area of expertise. Agile team members meet for face-to-face discussions on a regular basis to share their expertise and learn from each other:

> *"They set up some special interest groups for sharing knowledge within the company such as testing community, and developer community."* - P15, Agile Coach.

Chat Tool A chat tool is another discussion platform that facilitates the distribution of expertise. Agile team members have the ability to communicate by sharing their expertise at any time:

> *"We got Internet Relay Chat (IRC). It is an in-house chat [tool] for chatting within a specific group such as testers' group, developers' group, with different channels. This place is where we can share the information."* - P17, Software Tester.

> *"HipChat is really good. It is a team online chat like Skype."* - P46, Agile Coach.

Knowledge Sharing Session Through knowledge sharing sessions such as learning days and open space, Agile teams have a great opportunity to share their expertise. For instance, one of the participants mentioned that his organization runs a learning day for every sprint to foster a knowledge sharing culture in Agile teams:

> *"We have a learning day. Every sprint, we have a learning day. So we make activities for sharing knowledge."* - P39, Agile Coach.

Agile teams also adopt successful conference or seminar programmes in an Agile working environment. For example, open space has been applied in Agile teams as a platform in sharing knowledge and skills:

> *"We have an open space for about four hours. The first 30 minutes, we suggest the session. We have four slots and people can join the session. Open space is quite similar to open jam."* - P39, Agile Coach.

Robust Debate Surprisingly, a few participants (eg. P4 and P31) noted that it is normal to have robust debates when they were collaborating:

> *"We worked on the story. We had quite a lot of debate. If you look [from] outside, it might look like we are arguing and screaming at each other. But look inside, it is just a reflection of passion. All the time the idea is moved."* - P4, Agile Coach.

> *"Usually when the developers debate or disagree, I will let them do it. Even if the communication is bit heated, as long as they keep the professional level, I think it is fine. Sometimes, they need some fire to get better."* - P31, Scrum Master.

The aim of robust debate is to reach a resolution by finding convincing arguments. Open and genuine debates allow the distribution of expertise, particularly in solving problems and making decisions.

5.2.5 Archiving Explicit Knowledge

The previous approaches emphasize distributing tacit knowledge compared to explicit knowledge. As previously mentioned (see section 2.1), it is important to externalize tacit knowledge into explicit form for supporting expertise coordination. Therefore, another concept *archiving explicit knowledge* has emerged to describe how explicit knowledge can be disseminated, shared, and preserved in Agile teams through *whiteboards, videos* and *document management tools.*

Whiteboards The findings of this study revealed an interesting point, regarding an issue of being a hero in a team:

> *"We have got one resource here, who is being a single point of success for our team. He had to start coaching people and he needed to start spreading his knowledge. But nothing was happening because he had no time. He was too busy being a hero, fighting and fixing things. He could not spread his knowledge out."*- P22, Scrum Master.

As many problems occurred when relying on the only single expert in a team, a whiteboard was used as an initial stage in externalizing the expert's knowledge into explicit form:

> *"We asked the team to list at least 10 things that they needed to know on the whiteboard. So this guy [who is an expert] had to teach all the items. That was relatively successful and made some progress."*- P22, Scrum Master.

Sketching on whiteboards also facilitates Agile team members to externalize their mental models of expertise. The tacit knowledge can be visualized in explicit form through the sketch. The sketch has the ability to support the face-to-face communication in strengthening the distribution of expertise:

> *"That's very much like having some design sketch on a whiteboard. We need the sketch to understand and communicate how to do the code [sic]."* - P2, Agile Coach.

Video Videos can deliver knowledge and skills among Agile team members. Replaying the same video content for different audiences tends to save time and cost:

> *"You just point the video and let them watch the video. If you have 100 people, then 100 people can watch the video. Everybody can have the same understanding [sic]."*- P2, Agile Coach.

Document Management Tools Two document management tools have been identified from the findings of this study: Google Docs and Wikis. Google Docs allows Agile team members to create, edit, and share documents online. One participant claimed that he shared documents not just for sharing the software project information, but also for disseminating individual knowledge and skills. Everyone has the opportunity to access and update the shared documents in order to distribute expertise:

> *"For us as testers, we have spreadsheets on Google Docs for [saving] command lines. We share the command lines that we know on the spreadsheets, which are accessible for everyone."* - P17, Software Tester.

> *"We also use SharePoint. We can add folders, add projects, and each project has its own site. All documents can be uploaded to that site. Everyone can access and of course we can give access to everyone."* - P33, Project Manager.

Most participants (eg. P2, P16, P24, P31, P33, P42, and P46) argued that wikis are widely used in Agile teams for sharing knowledge. A strategy used by one of the participants, P2, for orientation purposes was sharing tribal knowledge through wikis:

> *"Now, many teams use wikis. Wikis actually report on some tribal knowledge...if the new developer joins, then they use this as induction [sic]."* - P2, Agile Coach.

Tribal knowledge is undocumented knowledge normally embedded in an expert's memory, which is very hard to retrieve, unless through an effective transmission of knowledge. Failure in retrieving tribal knowledge could cause many issues in accomplishing tasks. For example, Participant P42 claimed that:

> *"He had been on the team for a couple of years. He knew the product very well. There was no testing documentation. So, his tribal knowledge was returned to the history, and was really important for the team to success. He had the breadth of understanding of the products. He was able to be very effective and helped the team to solve any issue. When I joined the team, it was very difficult for me to get started because I didn't have any reference."* - P42, Software Tester.

Wikis allow the transition of tribal knowledge from undocumented knowledge to written knowledge. Through Wikis, Agile team members can share information that should be known by a newcomer before they engage in software development projects. Wikis play an important role in preserving tribal knowledge and supporting organizational memory.

5.3 A Relationship Between Categories of *Expertise Coordination Process*

Most of the findings of this study have shown the existence of the steps in sequence as depicted in Figure 5.8, where *locating and recognizing expertise* is the first step and is followed by *accessing expertise*. In certain circumstances, however, Agile teams need to share their expertise through *accessing expertise* before *locating and recognizing expertise*. For instance, Agile

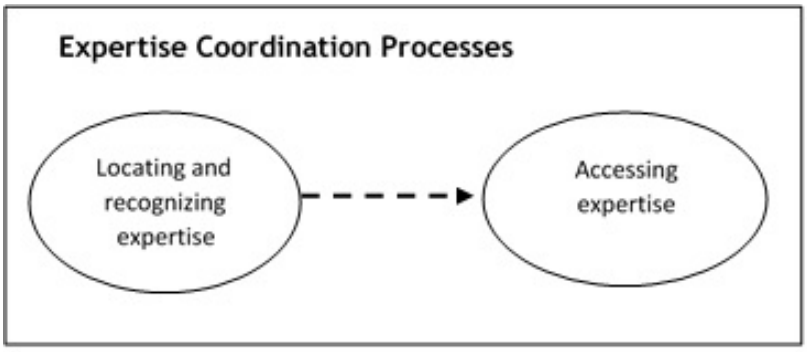


Figure 5.8: Relationships between *Locating and Recognizing Expertise* and *Accessing Expertise*.

team members have to demonstrate their work to others in order to gain recognition of their expertise.

Our findings revealed several approaches to *locating and recognizing expertise* and *accessing expertise*. Even though every step has its own approaches, we found common features in *locating and recognizing expertise,* and *accessing expertise* as follows:

- Both steps rely on effective communication and working closely together for successful expertise coordination. Although *communicating frequently* and *working closely together* are parts of *locating and recognizing expertise* approaches, these approaches indirectly facilitate the emergent approaches of *accessing expertise.*

- Agile practices such as daily-stand up and retrospective meetings foster the expertise coordination process.

We formulated a hypothesis based on a relationship between *locating and recognizing expertise* and *accessing expertise* as below:

H1 : *locating and recognizing expertise will have a positive influence on accessing expertise.*

The hypothesis was formulated based on the relationship between the emergent categories (see section 9.1.1.4). This hypothesis was stated in a positive manner to indicate a positive relationship between categories. For example, in the relationship between *locating and recognizing expertise* and *accessing expertise*, we found that *locating and recognizing expertise* tends to facilitate *accessing expertise*; Agile team members can easily retrieve available expertise once they know the owners of that particular expertise. Instead of possessing a massive amount of knowledge, every Agile team member just needs to know the owners of specific expertise. Once a team member needs certain expertise, they can identify the person who possesses that expertise and can access the expertise by relying on that person.

6

Coordinating Expertise Outside Agile Teams

This chapter describes how Agile teams manage and utilize outside expertise. The findings of this study revealed two different types of outside expertise that are involved in coordinating expertise: *external specialists* and *outsourcers*.

An **'external specialist'** is someone from outside the Agile teams but within the same organization, such as user experience designers, database administrators, and software architects. The external specialist is responsible for supporting Agile teams by bringing specialized skills. Agile teams can rely on individuals, parts of teams, or whole teams of external specialists. Each Agile project requires different types and numbers of external specialists depending on the project size and team composition.

Some companies also use outsourcing as a solution to fill in the skills gap or staff shortages. **'Outsourcers'** can be either an individual or a group of people from external companies, who provide their expertise to Agile teams for a certain period of time depending on contract agreements. Their

roles vary depending on the Agile teams' requests and are not restricted to Agile roles only. Table 6.1 summarizes the differences between external specialists and outsourcers.

Table 6.1: The Differences between *External Specialists* and *Outsourcers'* Characteristics.

External Specialists	**Outsourcers**
internal/external individual or teams	external individual or teams
support Agile teams depending on requests or throughout the whole project	fill in the expertise gap or staff shortages
specialist supporting roles	Agile roles or specialist supporting roles
working on a permanent basis	working on a contract basis

Based on the findings of this study, the reasons for relying on outside expertise in Agile teams are twofold:

Coping with new technology and paradigm Rapid growth in software technology involves a variety of frameworks, platforms and tools. The pace of change requires Agile teams to seek new skills and knowledge. When there is no in-house expertise, some companies rely on outside expertise as a solution for gaining new expertise:

> *"There are a lot of external new things that come along such as new frameworks, architectures, and tools. It is really difficult to have a workforce with those skills. Unfortunately, we are relying more on third parties to do all that work. We don't have the skill set internally and getting new skills is harsh."* - P24, Scrum Master.

New software technology requires more roles in Agile teams, which requires specialized expertise:

> *"Due to large products, our company requires specialization in a specific area. There are different browsers, tablets, mobile phones, and [software] requirements are getting bigger [sic]. Then, more [job] positions are required in Agile projects."* - P20, User Experience Designer.

In order to cope with new technology, some participants used outside expertise as a platform to learn new knowledge and skills:

> *"When we bought new technology, we hired a contractor to learn that technology."* - P26, Team Leader.

Agile teams sometimes face difficulties shifting to a new paradigm for developing software, including new practices, tools, and techniques. Consultants can be hired to assist Agile teams to change to a new paradigm:

> *"..... to change from the data warehouse, I had somebody come as a consultant to change our direction. The consultant has changed my mind and changed the direction of our project. So, the consultant was good for bringing a paradigm shift and convincing us to work in a new way."* - P26, Team Leader.

Changing teams' paradigms can require bringing new skills and knowledge into teams through external specialists and outsourcers.

Insufficient capacity of staff Failure in forming teams with the right capacity of staff tends to cause staff shortages. Outsourcing is often the fastest solution to fill the gap:

> *"When we are not staffed appropriately or we don't have the skill set to accomplish the project, we need consultants."* - P36, Developer.

The need for outside expertise depends on the size of software project. A big software project is often complex and requires more staff capacity. The findings of this study indicated that Agile teams often use outsourcing for testing the software due to insufficient staff:

> *"The biggest problem at our site is we don't have enough software testers. We have expertise in testing but the testing department is very new. There are only a few staff members and they need to work on many projects. We need more testing resources."* - P29, Developer.

The category *Coordinating Outside Expertise* emerged from the data analysis to describe how Agile teams, external specialists and outsourcers depend on each other to manage and utilize expertise outside teams. This category emerged based on the interviews of five external specialists: user experience designers, devops engineers, and one software architect. This category also emerged from the interviews with outsourcers, who had roles as developers and testers. These findings were also based on some Agile practitioners' perspectives of their experiences dealing with outside expertise.

The findings of this study revealed six strategies of *coordinating outside expertise*: *planning ahead (S1), understanding Agile mindset (S2), ensuring consistency (S3), retaining external expertise (S4), treating outsourcers the same as in-house staff (S5),* and *keeping everyone on the same page (S6).*

The relationships between the strategies of *coordinating outside expertise* are summarized in Figure 6.1. The oval shapes represent strategies, and arrows represent relationships between strategies. The figure depicts similar strategies involved in coordinating external specialists and outsourcers' expertise: *planning ahead (S1), understanding Agile mindset (S2), ensuring consistency (S3), retaining external expertise (S4),* and *keeping everyone on the same page (S6)* are relevant for coordinating both external specialists and outsourcers' expertise. A strategy of *treating outsourcers the same as in-house staff (S5)* is specifically applicable for coordinating outsourcers' expertise in Agile teams.

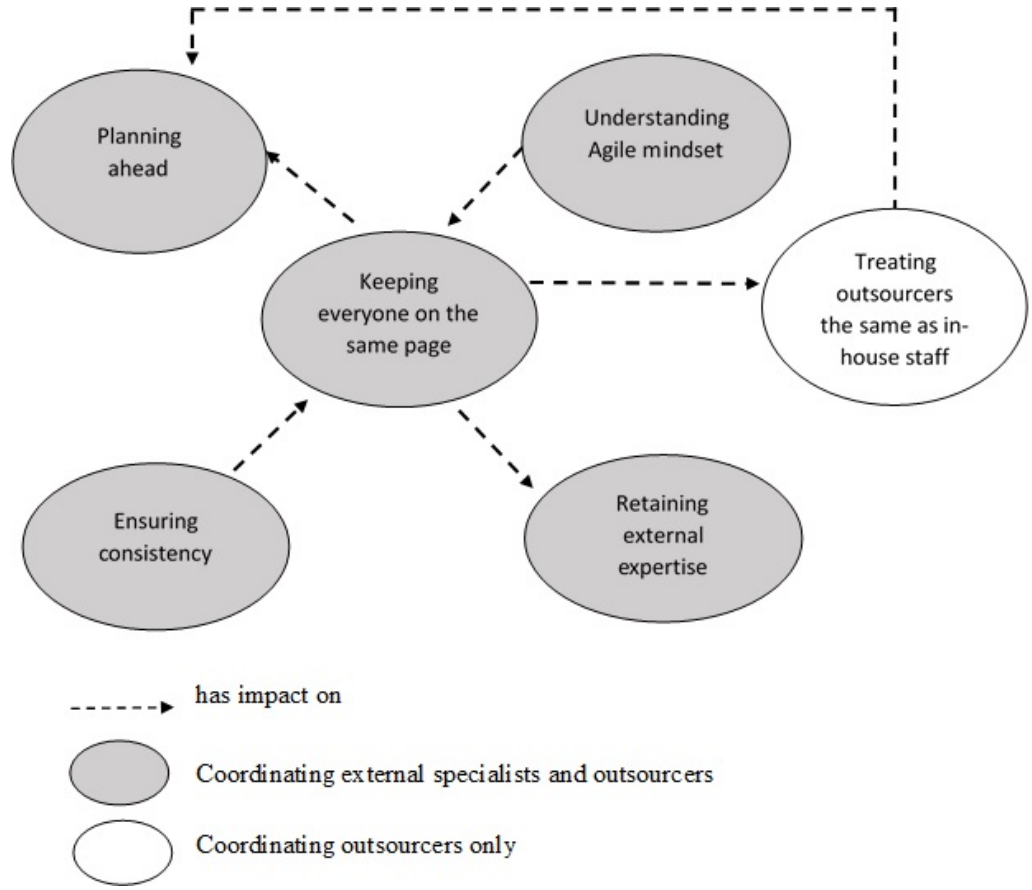


Figure 6.1: The Relationships between Strategies of *Coordinating Outside Expertise*.

6.1 Strategy 1: Planning Ahead

A strategy of *planning ahead (S1)* is used by teams to address availability issues that exist when Agile teams deal with outside expertise. Availability refers to the ability of external specialists and outsourcers to be present in Agile teams when their expertise is needed. The availability of external specialists and outsourcers in Agile teams depends on their workload and type of involvement. Workload is the amount of work assigned to external specialists in a specified period of time. The more work assigned to external

specialists and outsourcers, the more time they need to spend with the Agile team:

> *"It depends on how much we need them. Sometimes, we need full-time experienced designers to develop front-end applications, which involves a very intensive design."*- P41, Agile Coach.

Agile teams need to decide how to engage with external specialists and outsourcers. There are two ways external specialists and outsourcers can be involved in Agile teams:

- Throughout the whole project:

  > *"The DBA is actually involved throughout the life cycle of the project."*- P19, Business Analyst.

- Upon request:

  > *"I'm providing some sort of consultation. The team will ask me questions when my Agile team needs me."*- P44, Software Architect.

Most participants (e.g. P12, P18, P27, and P33) claimed that availability of external specialists and outsourcers are challenging for Agile teams. Agile teams had to rely on external specialists and outsourcers who were involved in multiple projects at one time. It was very hard for the external specialists to allocate their effort, responsibility, and time to the Agile teams. External specialists were sometimes unavailable when their expertise was needed. Such delays caused bottlenecks that affected the performance of teams:

> *"We have difficulties, such as the DBA is very busy and handles multiple projects. So to get his time can be more challenging."*- P33, Project Manager.

> *"He has a lot of other stuff to do. He has to do the database for other projects. So, he has to dedicate his time and skills when they are needed."*- P27, Developer.

The availability is also caused by a shortage of permanent staff for Agile Software Development projects.

> *"There is a shortage of permanent staff in Wellington, [particularly] for Agile scope. To find the right person with Agile [experience], it is still proving difficult in Wellington, especially permanent staff."*- P48, Tester.

Therefore, some software organizations choose to hire outsourcers for Agile Software Development projects. Unfortunately, hiring outsourcers involves more cost than recruiting permanent staff:

> *"It is expensive to bring in the contractor. I don't see everything as smooth but it's definitely challenging."*- P36, Developer.

> *"A lot of permanent staff are going to contracting. Because they know they do the same kind of work but they can be paid a lot more."*- P48, Tester.

Availability issues can be addressed through planning ahead, where Agile teams together with external specialists and outsourcers derive a shared vision of software development activities including tasks, workloads, and time frames. Planning ahead determines when external expertise is needed for a particular software project. Planning ahead requires Agile teams to discuss their plans with the external specialists or outsourcers to ensure the plan suits the external specialists' and outsourcers' availability. If there is an overlap with other tasks or projects, the Agile teams together with the external specialists or outsourcers need to adjust the plans to ensure the availability of external specialists or outsourcers:

> *"Usually, what we do is we plan ahead of the release. For example, we needed a solution architect for two or three months. So we tried to get them involved before that. We told them earlier when we needed them to be involved. Again it comes to planning. During planning, we get everyone to be involved. We make sure that they know what they need to do."* - P33, Project Manager.

Planning ahead relies on prioritizing tasks. Several participants (e.g. P12, P20, P35, P36, P41, and P44) indicated several criteria in prioritizing tasks. The value of the task should be the most important criterion when deciding which project needs to be worked on. The workload and period of time to accomplish the tasks should also be considered when prioritizing the tasks:

> *"Even they are outside resources that we depend on, we need to figure out what is the priority of the task compared to the other tasks."* - P36, Developer.

Figure 6.1 shows the relationship between *planning ahead (S1)* and *keeping everyone on the same page (S6)*. *Planning ahead (S1)* depends on *keeping everyone on the same page (S6)* between Agile teams and external specialists, as well as outsourcers. This relationship is discussed in subsection 6.6.

6.2 Strategy 2: Understanding Agile Mindset

A strategy of *understanding Agile mindset (S2)* is needed to be considered when engaging external specialists. External specialists and outsourcers may have a range of different software development backgrounds including Agile and non-Agile approaches. Some participants (e.g. P12, P18, P26, P27, P35, and P44) reported that they have to deal with external specialists and outsourcers who are unfamiliar with Agile methods. External specialists and outsourcers who are unfamiliar with or refused to apply Agile practices often work in different ways:

> *"I think it is hard because it is an external company. They got their way [sic] and we got our way in doing things. We tried to coordinate with something and it was quite difficult."* - P18, Developer.

It is hard to coordinate tasks when Agile teams and outsourcers work in different ways. Consequently, many problems will occur and cause project delay and cost overrun:

> *"The tester doesn't use Agile methods, and it's very hard for her to turn up for Scrum meetings."* - P23, Developer.

> *"That varies on contractors. It is not necessary for them to know about Agile and a lot of them don't. Some of them, if they have enough autonomy and authority, will resist and they won't be Agile. They will do things in their own way."* - P26, Team Leader.

> *"They intend to do things in their own way. The contractor often brought the project into trouble with lateness and bringing more resources."* - P26, Team Leader.

Some external specialists refused to learn and apply Agile practices, presumably because they did not see the importance of Agile values. This leads to Agile teams facing many problems when dealing with non-Agile external specialists. For instance, some external specialists were unable to align work with the sprints which required continuous value delivery at the end of each sprint. As a result, the external specialists failed to produce what they were expected to deliver for Agile teams on time:

> *"They [external specialists] didn't go to the same Agile training that we went through. We started doing our project, and we called our Database Administrator. We need these tables to be set up in two weeks. But it didn't go very well."* - P35, Developer.

Consequently, external specialists had to carry over the unfinished tasks to the next sprint and this delayed the next tasks. This situation became worse when the Agile teams could not continue their work due to dependencies on external expertise. Therefore, the Agile teams faced a problem in managing and coordinating rescheduled tasks. Without proper organization of tasks, it was hard to keep track of the tasks' progress:

> *"Then, the way we receive the requirements is like a waterfall model. They won't accept iterative release for the product. We quite often postpone and move the current stuff to the next sprint. That's not enjoyable. Sometimes we also couldn't believe what we had achieved or what we didn't achieve for the last sprint. That's the problem of continuity. There is a start and stop for several times [sic]."* - P27, Developer.

User Experience Designers are another kind of external specialist that typically prefer to produce a comprehensive user interface design before implementation begins. As Agile practices undertake relatively little up-front design, this has a big impact on the Agile teams, particularly the lack of feedback from the development side in improving the design:

> *"The User Experience Designer is the last batch of the waterfall approach [sic]. They don't know about the Agile method. They prefer the up-front [design] and not to share their work until it is finished [sic]. We have problems with that."* - P12, Agile Coach.

Agile approaches are very personality driven and require particular personality traits in Agile teams. Several participants claimed that they struggled to deal with some sorts of external specialists' misbehaviour:

> *"We had to deal with two DBAs. From time-to-time, we invited one of the DBAs for our stand-up or sprint review. But he just came once a week. It was because of the personality of the guy, who was quite difficult to deal with."* - P27, Developer.

Struggling with misbehaviour also happens the other way around. In certain circumstances, external specialists also have to deal with personality conflicts between Agile team members. One participant reported that he had faced difficulties when dealing with an Agile team member who refused to compromise in doing tasks together:

> *"I can't stand some behaviours. There might be a situation where a person holds a particular position for so long. There is no give and take."*- P44, Software Architect.

Coordinating outside expertise relies on the ability of Agile teams and external specialists, as well as outsourcers, to work in a cooperative manner. The progress of the project will suffer when cooperation is disrupted. It is very hard to develop good cooperation between Agile teams and external parties when personality conflicts happen in Agile projects.

Coordinating outside expertise relies on the ability of external specialists and outsourcers to align their work with Agile values and practices. In order to be familiar with and understand Agile values, it is vital to embrace *understanding Agile mindset (S2)*, which assists external specialists and outsourcers to work with the expectation of Agile teams.

Some Agile teams provide information about the Agile method to the external specialists through meetings, discussions, training, or workshops. This step facilitates the external specialists to be mentally prepared before engaging in Agile teams:

> *"We have explained a lot of times upfront to [the external specialists] who have signed up for the team. We discussed with them what and how we're doing in Agile [ways]. So they knew what they are responsible for and when it needs to be done. So once we started doing that, they got more ideas about the Agile [method]. Once they started to understand, then they started to communicate and learn more. It looks better now."* - P44, Software Architect.

One of the participants claimed that he joined his Agile team without a basic understanding of Agile methods. He consciously applied the Agile practices once he immersed himself in Agile teams:

> *"Yes, I hadn't discovered Agile [methods] and started working with Agile teams. I consciously used Agile [methods]. Knowing Agile is not compulsory. You can find a person who doesn't know anything about Agile [methods], but can act in an Agile way."* - P44, Software Architect.

It is quite challenging to deal with external specialists and outsourcers who are reluctant to accept Agile methods. One of the participants (P26) indicated that he did not force them to work in an Agile way. He gave them authority to apply their own software method. From time to time he and other Agile team members tried to educate the external specialists and outsourcers by showing the benefits of Agile practices. Through good behaviour that had been demonstrated by Agile team members, there is a possibility that external specialists and outsourcers may change their mind-set into Agility:

> *"If they have enough autonomy and authority, they will resist and won't be Agile [people]. They will do things their way. Some people don't believe in Agile methods. If anybody refuses to apply the Agile method, we have to respect them. They have their experience, and the way they are doing things. As the project progressed, when something went wrong, I said, 'Hey look, this is why we do the Agile way.' "* - P26, Project Manager.

As Agile methods emphasize people and interaction, people skills are essential resources for Agile projects. People skills are hard to gain without the ability to interrelate with others. Thus, people skills can be developed through an awareness of how to interact with others and practising those skills in the right ways:

> *"The most important things are they can talk and communicate. Those are people skills that they should have more than anything else. They can do their jobs but they should be able to interrelate with people. If they can't interrelate with somebody, I don't think Agile works for them."* - P4, Agile Coach.

Educating external specialists to act in an Agile way requires a willingness from both parties, Agile teams and external specialists. The Agile teams need to understand the Agile method and behave in an Agile way. They also need to educate the external specialists and outsourcers about Agile values and practices. The Agile teams need to know and apply suitable methods for educating the external specialists and outsourcers about Agile methods. The perception and willingness of the external specialists and outsourcers to shift their paradigm into the Agile way, however, determines the success of educating them about Agile methods.

As depicted in Figure 6.1, *understanding Agile mindset (S2)* is interrelated with *keeping everyone on the same page (S6)* in coordinating outside expertise. Failure to understand Agile values and practices hinders external specialists and outsourcers from communicating and collaborating effectively with Agile teams. As a result, external specialists and outsourcers do not realize the importance of Agile meetings such as daily stand-up meetings or retrospectives in establishing effective communication and collaboration with Agile teams.

6.3 Strategy 3: Ensuring Consistency

The strategy *ensuring consistency (S3)* is vital when considering stability issues that arise in Agile teams. Stability refers to keeping Agile teams stable with a low rate of turnover of both team members and external specialists, as well as outsourcers. Stability is an important factor that tends to affect expertise coordination in Agile teams. Many problems arise when

there is high fluctuation in Agile teams including external specialist and outsourcer turnover.

Some external specialists reported that they have had to change to another team while they were still working on on-going Agile projects. They have had to adapt to a new environment with different specifications once they moved to a new team. It takes time for them to cope with the new team and indirectly affects their progress. Failure to adapt to the new team could cause them to be unable to produce the expected deliverables on time:

> *"Changing teams happens all the time. Frequent changes in the [Agile] project requires me to be flexible."* - P38, User Experience Designer.

Involvement in unstable Agile teams has a negative impact on the external specialists and outsourcers. For instance, one of the participants, a software architect P44, claimed that he didn't see the benefit of his presence in the Agile team. It was impossible to get support from Agile team members to accomplish his tasks while the team was struggling to solve their internal problems:

> *"People move to another team regularly. Teams need to recover from the changes. The team can't get stable and gel together. So my role moves slowly. They are busy with other stuff, and they don't engage. Nothing is done. So my role becomes irrelevant because [the project] it is not progressing well. So, there is no point."* - P44, Software Architect.

Ensuring consistency is essential when dealing with stability issues. Participant P38 pointed out the need for a consistent standard of work. Coping with frequent external specialists turnover requires a consistent standard of work, which enables a new external specialist or outsourcer to easily adapt to the new team and work:

> *"Frequent changes in the [Agile] project require me to be very flexible. I have to familiarize myself as quickly as possible. So, we make sure that we do things more consistent [sic]. It is not difficult for someone to pick up the work."* - P38, User Experience Designer.

Figure 6.1 shows a relationship between *ensuring consistency (S3)* and *keeping everyone on the same page (S6)*. Communication will be disrupted when existing external specialists leave Agile teams and new specialists join teams. It is difficult to enable effective communication among Agile teams, external specialists, and outsourcers when there is a high rate of turnover.

6.4 Strategy 4: Retaining External Expertise

In coordinating outside expertise, *retaining external expertise (S4)* is important to capture external specialists' knowledge and preserve the knowledge within Agile teams. Retaining external expertise should be extended to every segment of Agile teams, including outsourcers:

> *"The things that got my bear there is a sustainability [sic]. How can the team learn from the person who comes in and then disappears? How can we extract the knowledge and learn from them?"* - P16, Agile Coach.

Several participants (e.g. P23, P26, P27, and P33) posited that they preferred working with in-house staff because team knowledge is retained in Agile teams:

> *"The permanent staff are easier because the knowledge stays here."* - P27, Developer.

Once the project is finished, outsourcers will leave the teams, taking their expertise with them. Many problems can occur if there is no effort to retain the outsourcer's knowledge:

> *"The outsourced staff are expert but the only problem is, once the project is finished, they will go. If they are not passing their knowledge to the internal staff, the maintenance and supporting work become hard. When they go, the knowledge will be lost."* - P23, Developer.

One of the participants, however, reported that time constraints hinder the knowledge sharing between outsourcers and Agile teams:

> *"They are very good but to a certain extent they find it very hard to delegate their time to teach a particular guy."* - P23, Developer.

Retaining outsourcers' expertise depends on roles and the project's needs. Agile teams need to decide which outsourcer's expertise has a significant impact on future tasks and projects. For instance, more priority is put on developers than testers since the developer's knowledge is important for maintaining the software:

> *"We have people from outside work with us, such as testers. The whole testers, the testing team, we outsource. There is no issue in sharing expertise with them. We have a kind of problem with the developers that we are outsourcing. They normally take their knowledge away, as the project is finished. That can be quite challenging if we want to do software maintenance. But for testers, it is more like testing, finding bugs, quality issues, and there is no challenge."* - P33, Project Manager.

The same goes for external specialists. Agile teams need to decide which external specialists' expertise has a significant impact on other roles. For instance, sharing software design ideas with developers assists developers to implement the software. This tends to speed up the development of software project:

> *"We help developers to understand the design principles and get them to sketch with us."* - P38, User Experience Designer.

Agile teams rely too much on external specialists when they fail to retain external specialists' knowledge. These dependencies tend to affect and delay other tasks. Thus, it is essential to transfer external specialists' knowledge to Agile team members. If external specialists are unavailable, at least someone in the team needs to be able to troubleshoot simple and routine problems.

> *"If possible we try to make sure the DBA can share his work and gain a new skill. So we try not to rely so much on him."* - P39, Agile Coach.

Retaining external specialist and outsourcer expertise in Agile teams is vital for long-term retention on external expertise. Failure to sustain external specialist and outsourcer expertise will affect the team's knowledge in the long term. If the expertise is not retained, Agile teams have to continue to rely on outsourcers for maintaining the software or committing to the next project.

For long-term investment, retaining external specialist and outsourcer expertise is vital for managing the next task and new projects. It is impossible to retain all external specialist and outsourcer knowledge and skills. Thus, Agile teams need to decide which expertise needs to be retained and how to retain it:

> *"But at the end, some knowledge will walk out the door with the contractor. We can't guarantee every skill that the contractor has can be passed on to the staff."* - P26, Team Leader.

There are several approaches for retaining external specialist and outsourcer expertise in Agile teams, such as pair-programming, documentation, and mentoring (refer to Chapter 5):

6.4.1 Pair-programming

Several participants [eg. P26 and P33] mentioned the use of pair-programming in transferring outsourcers' expertise into Agile teams. Pair-programming emphasizes the development of skills among Agile team members. Once Agile teams can grasp the skills, there is probably less dependency on outsourcing in future:

> *"Other than that, we also use pairing, and the contractor is paired up with the developer."* - P26, Team Leader.

> *"We need to do a lot of knowledge sharing, pair-programming, between internal and external people."* - P33, Project Manager.

6.4.2 Documentation

As opposed to pair-programming, which gains expertise tacitly, documentation is used to capture outsourcers' knowledge explicitly. This method is applicable as references for managing future tasks and might be for future projects.

> *"If we have consultants, we have to make sure their knowledge can be passed on to the permanent staff. It might be through documentation."* - P27, Developer.

6.4.3 Mentoring

Outsourcers can act as mentors in facilitating Agile team members to gain new knowledge and skills:

> *"If we have consultants, we have to make sure the knowledge can be passed to the permanents staffs, might be through documentation..... We also pass on knowledge by mentoring."* - P27, Developer.

With the proper guidance and adequate expertise of outsourcers, Agile teams can gain benefits through the mentoring approach:

> *"We engage with external consultants to define the design language for our product. It will be our design guidelines. Then, we will use the design guidelines to influence our general design. So, the product owner has to pick up the skills for designing the user interface."* - P31, Scrum Master.

Willingness to share knowledge is a basis of retaining outsourcers' knowledge in Agile teams. Most participants (e.g. P12, P23, P26, P27, P31, P38, and P39) indicated a positive knowledge sharing culture between outsourcers and Agile teams. As many opportunities for work are available for outsourcers, there is nothing to lose when outsourcers pass their knowledge and skills to Agile teams:

> *"In the past I had contractors who had developed the software and gave it to us, that's it. But most of the time I see, especially in Wellington, where there is too much work for contractors here. They are not really afraid to share their knowledge about their work, because they know so much work is coming."* - P24, Scrum Master.

Figure 6.1 shows the relationship between *ensuring external expertise (S3)* and *keeping everyone on the same page (S6)*. *Ensuring external expertise (S3)* depends on *keeping everyone on the same page (S6)* between Agile teams, external specialists, and outsourcers. This relationship is discussed in subsection 6.6.

6.5 Strategy 5: Treating Outsourcers the Same as In-house Staff

A strategy of *treating outsourcers the same as in-house staff (S5)* is important to bridge the gap between outsourcers and Agile teams since they are new

to Agile teams and have little time to adapt to a new working environment. Coping with a new environment requires outsourcers to understand the software development process. Outsourcers need extra time to learn the software domain before engaging in their work:

> *"I don't see everything as smooth but it's definitely challenging, because they don't understand the application. They are new to the company. So, we have to give them some time to learn about the application."* - P36, Developer.

There is a negative impact on Agile teams if the outsourcers are unable to adapt to the new environment quickly and understand the software domain and application. This situation tends to affect the development progress of software.

In a new working environment, it will take time to get to know in-house staff:

> *"Knowing who is in the team, how they work and building that trust again, it is slow to begin"* - P48, Tester.

Consequently, a few outsourcers are overconfident in their field of expertise. This becomes worse when they are unable to recognize available in-house expertise:

> *"The contractors come in with a lot of self-confidence on their own. They are very confident in what they are good at. Sometimes, they don't necessarily see in-house staff who have expertise."* - P26, Team Leader.

The same thing goes for Agile teams in getting to know outsourcers. As outsourcers are new to Agile teams, it is vital to identify and recognize outsourcers' expertise. Hence, one of the participants (P24) claimed that he preferred working with in-house staff because they knew the available in-house expertise:

> *"I prefer permanent staff because in the next project I know who did good jobs in the previous project, and I know what skills they have."* - P24, Scrum Master.

It will take time to build trust between Agile teams and outsourcers. Different expectations in the level of expertise can cause difficulties in achieving mutual agreement between Agile teams and outsourcers.

As outsourcers are external parties, it is important to bridge the gap between them and Agile teams. One of participants treated outsourcers the same as in-house staff to assist the outsourcers to quickly adapt to an unknown working environment:

> *"A diversity of people is always good. The contractors come in and give us a certain thing [expertise], but we have to manage them as permanent staff."* - P26, Team Leader.

Agile teams can help outsourcers feel like they are part of teams through frequent contact and engagement:

> *"If we hire contractors, we have to make sure they feel they are a part of the team. Joining the team, staying together, participating in meetings, and making shared commitments."* - P39, Agile Coach.

> *"We need to have this person be a part of the team. We need the contractor to feel like we are his team. We have to generate frequent contact to ensure he feels part of the team. He has to be involved in our meetings, including planning meetings."* - P39, Agile Coach.

Besides attending Agile meetings, there are several options for treating outsourcers as in-house staff:

> *"It can range from each organization, but usually the companies that I used to work with included contractors in some activities that were traditionally reserved for permanent staff only. For example, they included contractors in team-building exercises, team celebrations and also innovation days."* - P48, Software Tester.

Through frequent contact and engagement, Agile teams have opportunities to show their concerns, awareness, and appreciation of outsourcers' expertise. Indirectly, this could remind them of the value of the outsourcers' expertise and their availability for Agile teams. Therefore, there is a relationship between *treating outsourcers the same as in-house staff (S5)* and *planning ahead (S1)* as depicted in Figure 6.1. Appreciating outsourcers encourages them to be involved in every software development activity.

Figure 6.1 shows the relationship between *treating outsourcers the same as in-house staff (S5)* and *keeping everyone on the same page (S6)*. *Treating outsourcers the same as in-house staff (S5)* depends on *keeping everyone on the same page (S6)* between Agile teams and outsourcers. This relationship is discussed further in subsection 6.6.

6.6 Strategy 6: Keeping Everyone on the Same Page

The findings of this study revealed *keeping everyone on the same page (S6)* is a strategy to establishing effective communication. A strategy of *keeping everyone on the same page (S6)* involved a process of conveying sufficient explicit or tacit knowledge between Agile teams and external parties either verbally or non-verbally. Effective communication enables good cooperation between Agile teams and external specialists, as well as outsourcers. Poor communication leads to failure in coordinating outside expertise in Agile teams. Consequently, Agile teams, external specialists, and outsourcers tend to point the finger and place blame on each other instead of finding solutions:

> *"The developer and the operation staff didn't talk to each other. This operation team did the deployment. Sometimes, there was failure, and we found that the development team blamed the operation team. The operation team blamed the development team."* - P15, Agile Coach.

It is difficult to coordinate outside expertise when Agile teams convey incorrect and insufficient information to external specialists. Agile teams need to provide clear goals for the Agile software project at the beginning of the project. A lack of mutual goals drives the external specialists to work in their own direction without considering the whole project:

> *"It is hard because they have their ways and we have our ways in doing the tasks. We try to coordinate, but it is quite difficult."* - P18, Developer.

Conveying insufficient information about task descriptions causes external specialists and outsourcers to be unable to align work with Agile teams' needs and expectations. External specialists fail to perform when they are not really clear about their roles and responsibilities. They also do not know when they should be available for Agile teams. This becomes worse when they are unable to be present when their expertise is needed:

> *"They don't know what is happening if they don't have the project's visibility. They didn't know when they are needed, and they didn't have a feeling of being involved."* - P16, Agile Coach.

Furthermore, our participants reported that it is hard to make sure everyone including external specialists has a similar understanding of the project's progress development:

> *"Usually the issue is to make sure that everybody is on the same page."*
> - P38, User Experience Designer.

These problems happen due to ineffective communication between Agile teams and external specialists. Agile teams should provide sufficient information to external specialists, verbally or in writing. On the other hand, external specialists also need to provide Agile teams with necessary information such as their availability, needs, and expectations when dealing with Agile teams.

Through Agile meetings, Agile teams have opportunities to show their concerns, awareness, and appreciation to external specialists and outsourcers. Indirectly, this could remind Agile teams of the value of the external expertise, and their availability:

> *"For every project, we have different databases and architecture. So we get somebody, and we pull them into a sprint. We make them a part of the team. Bringing them to the daily meetings, reinforcing them and making them realize 'Aah, this guy is true. He is waiting for me. That's why he keeps reminding me.' "* - P35, Developer.

Working closely together depends on the workload that external specialists and outsourcers need to contribute to Agile teams. For a high volume workload, it is better for external specialists and outsourcers to stay and work closely with Agile teams. For a minimal workload, however, external specialists and outsourcers need to figure out how long they need to allocate their time for Agile teams:

> *"Depends on how much the work is [sic]. If there is a lot of work, we ask them to move their stuff and computer, and come over to us."* - P12, Agile Coach.

Agile teams should provide necessary information without overwhelming the external specialists and outsourcers with an overload of information. Too much information may lead to more confusion than clarity and cause misunderstanding between Agile teams and external parties:

> *"So the key thing is to keep them to be involved and keep them to be informed through the process. We also share the information on wikis. It is tricky and not easy, but we keep informing them as much as we can without swamping them."* - P16, Agile Coach.

The findings of this study indicated that *keeping everyone on the same page (S6)* is the centre of the strategies' relationships. Figure 6.1 in section

3.2 shows the relationships between *keeping everyone on the same page (S6)* and other strategies. *Keeping everyone on the same page (S6)* tends to affect *planning ahead (S1)*, *retaining external expertise (S4)*, and *treating outsourcers the same as in-house staff (S5)*. The relationship between *keeping everyone on the same page (S6)* and *planning ahead (S1)* allows conveying sufficient information to external specialists, which assists them to bring the right and relevant expertise to teams in a timely manner. In terms of *retaining external expertise (S4)*, *keeping everyone on the same page (S6)* facilitates knowledge transfer from outsourcers and external specialists to Agile teams. *Keeping everyone on the same page (S6)* also tends to bridge the gap between Agile teams and outsourcers and indirectly quickly adapt to *treating outsourcers the same as in-house staff (S5).*

Keeping everyone on the same page (S6) depends on *understanding Agile mindset (S2)* and *ensuring consistency (S3)*. Failure to understand Agile values and practices hinders effective communication happens between Agile teams and external parties. *Ensuring consistency (S3)* also affects the *keeping everyone the same page (S6)* because it is impossible to establish effective communication if Agile teams or external specialists change often.

Therefore, we formulated six hypotheses based on relationships between categories of *coordinating outside expertise* as follows:

- H16 : *keeping everyone on the same page will have a positive influence on planning ahead*.

- H17 : *keeping everyone on the same page will have a positive influence on retaining external expertise*.

- H18 : *keeping everyone on the same page will have a positive influence on treating outsourcers the same as in-house staff*.

- H19 : *understanding Agile mindset will have a positive influence on keeping everyone on the same page*.

- H20 : *ensuring consistency will have a positive influence on keeping everyone on the same page*.
- H21 : *treating outsourcers the same as in-house staff will have a positive influence on planning ahead*.

7

Management Support

This chapter decribes how management support influences expertise coordination for Agile Software Development projects. There are three strategies of management support that emerged from the data analysis in supporting expertise coordination:

- Self-Selecting Teams
- Reforming Performance Appraisal
- Embracing Expertise Sharing Culture

The details of strategies are summarized in Figure 7.1. The next subsections describe each strategy involved in supporting expertise coordination.

7.1 Self-Selecting Teams

This study indicated that Agile teams are expected to be self-organizing teams starting with the initial stage of forming teams. Based on the findings

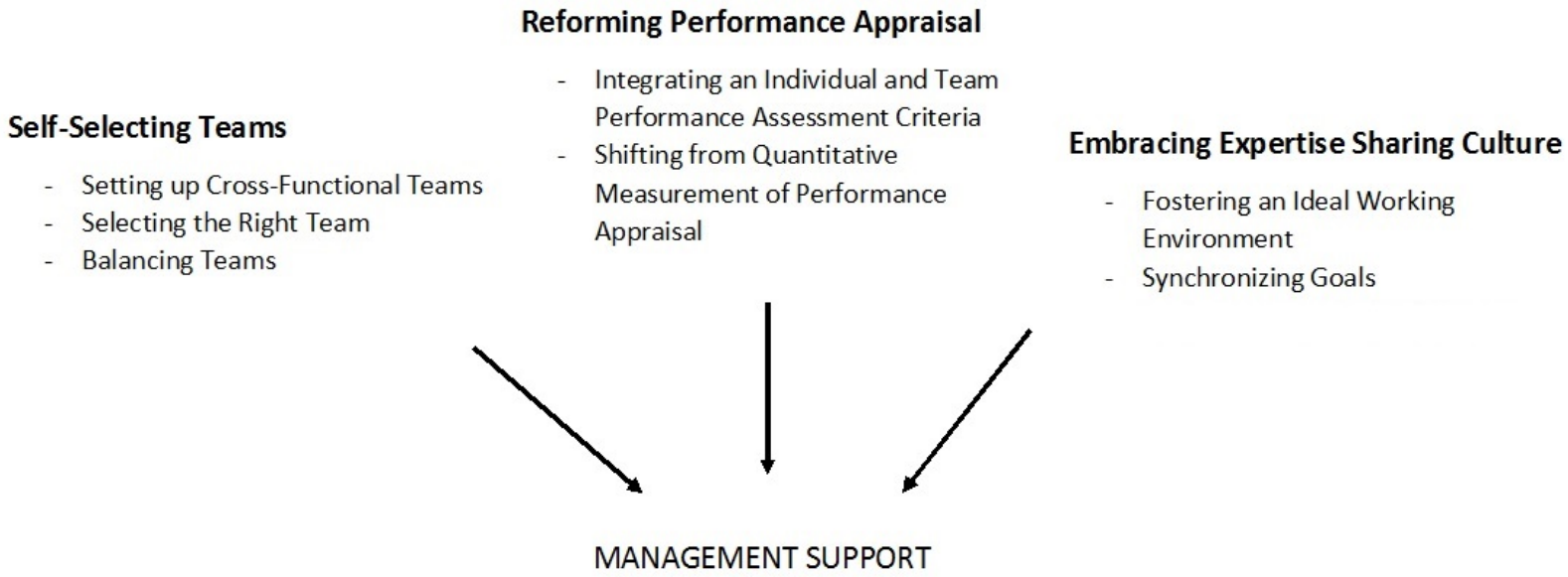


Figure 7.1: Management Support in Coordinating Expertise.

of this study, however, traditional team formation is still being applied in some Agile organizations. The management of the organization decides the membership of the teams based on the expertise required for the software project:

> *"That's the ideal because the team is self-organizing and they know each other and they know they can fix the problem. But that's not the real world. The team is made up by the vendor. The vendor will identify the people that they need. Or management has a meeting and there is a kind of human resources games."*- P22, Scrum Master.

> *"Our team leader decides who will be in the team based on our expertise."*- P23, Developer.

The starting point to embrace self-organizing teams is through self-selecting teams, which enable Agile team members to form their own teams. Forming the self-selected team involves interdependence between Agile team members. Therefore, the power of self-selecting teams can be seen in the way Agile team members effectively bond together:

> *"The teams are self-selected. We run the selection based on the individual exercise, to allow people to pick which squad they want to work with. So, they pick a squad that can stay together and work together. They know each other very well."*- P46, Agile Coach.

The findings of this study revealed three activities required in facilitating self-selecting teams for Agile Software Development projects as follows:

- Setting up cross-functional teams
- Selecting the right team
- Balancing teams

7.1.1 Setting Up Cross-functional Teams

Cross-functional teams include different functional expertise required to develop software (see section 2.3.3). The team is set up based on a business domain area, which enables the management of the organization to set and clarify goals and objectives for each team. The management can then define the roles required for each team. Based on the findings of this study, different skills are required in a team in order to fill basic roles. The roles might change depending on requirements of the software project:

> *"We have a changeable blueprint. So, we think the skills are business analyst, designer, developer, tester and some kind of infrastructure production. We think all squads, even though they vary, need those skills. Each squad also needs a Scrum master. Each squad also needs a product owner as well to define what we are going to build. Generally, it is about seven skills."*- P46, Agile Coach.

The number of team members required for each role varies depending on the project. Minimizing the team size, however, needs to be considered when setting up Agile teams:

> *"Each squad is made of three to seven people. In some cases, the biggest squad has one person per skill."*- P46, Agile Coach.

> *"The number of team members is different but there is a baseline. Most teams at least have two developers and testers. Each team needs a product owner or Scrum master. It is like a basic team. On top of that, depending on the nature of the team, we might have a designer and BA, but not for all teams. It depends on work."*- P47, Developer.

According to some participants (eg. P12, P21, P22, P32, and P24) a big team causes several problems that hinder successful expertise coordination. There will be more communication and interaction lines once more team members are involved, as well as a wide range of knowledge and skills.

As a larger number of team members are involved, however, it becomes harder to identify who has the sort of expertise needed and how to access the required expertise:

> *"We have a team which is bigger than nine people. We see lots of problems in sharing expertise [sic]. There are many different lines of communication and it is hard to coordinate."*- P21, Agile Coach.

> *"In the big team, the more interactions involved and the more assumptions made, the more problems need to be fixed later on."*- P24, Scrum Master.

The next major problem is the accessibility of team members. For instance, Participant P22 mentioned that his team members refused to attend daily stand-up meetings. A daily stand-up meeting provides a communication vehicle for Agile team members to raise issues and obstacles that impede their progress. As there are many members involved, it is hard to strictly monitor and manage every member. If they regularly miss meetings, more problems will occur including degrading expertise coordination. Failing to attend the meeting will affect the process of finding and divulging the necessary expertise and solution:

> *"They missed the stand-up meeting because it is too big."*- P22, Scrum Master.

> *"What I found in a small team is that it is easier to get everybody in the room and have everybody involved in discussion and conversation."*- P24, Scrum Master.

Therefore, a small team is important for Agile team members to bond together easily and directly induces awareness of team members' expertise and performance. Participant P22 indicated that a small team provides space for him to notice other team members who are facing problems

or obstacles in accomplishing their tasks. It is important to immediately identify those people in order to provide relevant expertise in solving their problems:

> *"When they are in the small team, they bond better, are dynamic and more manageable. It is also easy to see someone who is struggling for help."*- P22, Scrum Master.

An interesting point that has been revealed by one of the participants is the velocity of performance. A small team tends to speed up the process of forming the team and assists team members to quickly bond together. Therefore, a small team also tends to speed up the process of expertise coordination in Agile teams, including identifying, recognizing, and accessing relevant expertise:

> *"A small team is much easier to communicate with and we get much more velocity. Forming the team is quicker and we know each other and work together better."*- P24, Scrum Master.

It is important to properly set up Agile teams based on desired goals, objectives, and roles needed for each team. Failure in setting up the right Agile teams, particularly with an optimal size, tends to affect the next steps in self-selecting Agile teams.

7.1.2 Selecting the Right Team

Effective self-organizing teams require the management to allow employees to choose which software project they intend to work on and with whom they prefer to work. Before selecting a team, every potential Agile team member is provided with an overview of the software project involved, including the goals and objectives of each team. The purpose of this overview is to ensure they clearly understand their roles and responsibilities. Based on the information given, they will consider the diversity of skills and experience of potential team members when selecting the team:

> *"So, we provided information up-front, which is more details about what they will be signed up for. So, they can have some conversation before [selecting the team]. During the self-selection team, we work hard to make sure the right people were talking together. So, I and another coach will help to facilitate the conversation."*- P46, Agile Coach.

The findings of this study indicated several factors that participants (e.g. P6, P23, P46, and P47) tend to consider when selecting a team: types of work, team members, and knowledge sharing. It is common to select a team based on types of work or project involved:

> *"When everyone is selecting a team, they look at what type of work that they want to do."*- P47, Developer.

Every Agile team member knows their capabilities and strengths and how they can contribute to ensuring the project is success. They try to match their expertise with the requirements of the project when selecting their preferred team:

> *"I focus more on something that I can provide, my expertise and what type of project."*- P23, Developer.

In selecting a team, some Agile team members consider who they work with. Choosing the right team members will enable them to work well together:

> *"If they see someone's name and they think that they won't work well with that person, they will change to another squad."*- P47, Developer.

Participant P47, however, argued that Agile teams put priority on work instead of team members when deciding their preferred teams.

> *"Closeness to others in the organization definitely plays a part but not the main one [sic]."*- P47, Developer.

Surprisingly, Participant P46 pointed out that knowledge sharing is a pivotal factor when Agile team members select their team. They chose their preferred team based on what knowledge and skills that they can gain from the team and what they can contribute in enhancing team knowledge:

> *"There are two questions they asked themselves. One, 'what can I learn from this squad?'..... Second, is to be an expert. 'What can I teach this squad?'."*- P46, Agile Coach.

In order to enable knowledge sharing, a diversity of expertise and experience are essential for each team. An Agile team should consist of different roles, and sharing expertise exists between the same roles, and within and across different roles as well:

> *"We are not just sharing knowledge between developers, but also between developers and testers, testers and business analysts."*- P6, Agile Coach.

Based on our observation at company XYZ, we found active knowledge sharing happened between developers or between product owners, and between all team members as well. Hence, during team selection, team members consider how well they can work with and how well they can share knowledge together.

Besides role composition, mixing junior and senior team members in a team is best practice. Ensuring the diversity of expertise and experience, however, should reflect the needs and requirements of the software project:

> *"So, people want to join the squad with novices, beginners or juniors, because they want an opportunity to help them and coach them."*- P46, Agile Coach.

> *"As a senior, I have to work with junior staff. Because most of the time, I'm not probably producing something for myself but I teach others."*- P47, Developer.

Knowledge sharing creates a win-win situation, which allows juniors, as well as seniors, to gain benefits in sharing their knowledge and skills. From the juniors' point of view, they have opportunities to develop their individual expertise with guidance from seniors. Besides enhancing team knowledge, seniors also have the chance to be experts in their particular area of expertise and this indirectly tends to improve their coaching skills.

In terms of implementation, self-selection is quite difficult for new staff who have recently joined the organization. Based on the findings of this study, they took more time than their senior members to decide their preferred team. The main reason is how well they can work with others. Their curiosity towards other members' attitude, capability, and expertise urges them to make a decision for selecting a team:

> *"If they have been around a while, they could very quickly make a decision. If they have just joined this company, it is very hard to make that call. They don't know enough whether they can work well together."*- P46, Agile Coach.

Therefore, selecting the right team requires facilitation from coaches and management of the organization. Proper guidance from them will assist Agile team members to have a valuable space for conversation and get the right information before choosing a team. This will facilitate the right implementation of self-selecting teams, which tends to form the right team with the right team members.

7.1.3 Balancing the Teams

After selecting a team, the management of the organization and project leaders have to ensure well-balanced teams are formed. Each team should

be composed of the right roles and appropriate number of team members, which align with the needs and objectives of the team:

> *"So, everyone selects a team that they want to work for. At the end of the day, the team needs to be balanced. We can't have five developers here and one and only in the other team."*- P47, Developer.

Besides roles and number of team members, balancing background experience between junior and senior members also needs to be taken into account. Working with team members of the same level of expertise tends to hinder effective knowledge sharing. Compared to senior members, junior members tend to feel negative effects when they are left behind in developing their knowledge and skills without support and guidance. Therefore, it is important to balance levels of experience within teams, even though there is no specific rule in deciding the composition of the number of junior and senior members in a team:

> *"Some squads don't want too many junior developers in the team because it requires a lot of learning."*- P47, Developer.

Balancing the teams requires persuasive skills by the project leader and management of the organization. The finding of this study indicated that this technique was used to balance the teams. This provides an opportunity for Agile team members to consider the advantages and disadvantages for joining another team. The final decision, however, is still made by Agile team members in self-selecting teams:

> *"But when the team balance is not quite right, then the management and project leader will come to convince us to think about the other squad. Some people choose to do that after they convinced them to join the other team. They don't force us to move to the other squad. But, they tell us what is the best for the company."*- P47, Developer.

When balancing teams, an organization needs to consider how to ensure the teams are likely to be stable. Team stability requires Agile team members to remain intact for the long term, truly gel together, progress well together, and trust each other. The findings of this study revealed that self-selecting teams have a positive impact on team stability because little fluctuation in the team composition has been reported:

> *"Our squads are normally stable but there is some movement. If they decide to leave the squad, they need to have a conversation with the other squad."*- P46, Agile Coach.

Having stable teams can be beneficial for coordinating expertise in Agile teams. Little fluctuation in team composition enables the stability of a team's skill set. Team members do not need to frequently update their meta-knowledge of the available expertise in their teams:

> *"How can the team learn from the person who comes in and or the person who comes in right and disappears. If we get somebody else in, how can we extract the knowledge and learn from them? The skill set has to be sustainable [sic]."*- P16, Agile Coach.

Stable teams induce shared responsibilities to ensure the quality of the whole software development. Participant P2 pointed out that tight bonding between Agile team members enables them to be more responsible toward other roles. They tend to absorb other roles when their expertise is needed:

> *"They develop a good sense of maturity and run the team well and they can see roles in their team as their own."*- P2, Agile Coach.

The final activity of self-selecting teams, balancing teams, assists in improving the team structure with the right roles and experience within teams. Management and project leader involvement, however, are important for balancing teams with consideration and cooperation from Agile team members in shifting to the other teams.

Self-selection is a baseline in supporting expertise coordination in Agile teams. The right team with the right team members can be formed through self-selection, which assists Agile team members in sharing and accessing expertise when it is needed. Furthermore, minimizing team size, and balancing and stabilizing teams can be implemented through the sequence of self-selecting team activities, which have a positive impact on expertise coordination. Therefore, it is vital for the management of an organization to implement self-selecting teams in supporting expertise coordination.

7.2 Reforming Performance Appraisal

The findings of this study indicated the need for performance appraisal in identifying gaps in an employee's skills or competencies, as well as opportunities for improvement and development of related skills. As an Agile Software Development team is team-oriented, Agile team members need to rely on each other in developing their expertise and team knowledge. This is the focal point, where expertise coordination is needed in Agile teams for the development of skills.

Traditional performance appraisal is based on an individual assessment, which measures an employee's work against measurable objectives. This type of assessment focuses on the skills exercised in the current tasks and skills that must be acquired for the next project. For instance, one of the participants claimed that his performance appraisal was based on the point of velocity or amount of work that he can accomplish in a certain sprint:

> *"One organization that I worked at previously, the idea [of performance appraisal] was measuring developer's velocity. How many capacity points does the developer deliver per sprint?"* - P16, Agile Coach.

A major drawback of the traditional performance appraisal is gaming the system. The findings of this study indicated that some Agile team members tend to choose the easiest task or user story in order to perform

well in their team, without considering other team members' capabilities and expertise. There is a possibility for Agile team members to be selfish, inconsiderate, and intolerant in achieving their key performance indicators. These attitudes have a negative impact on teamwork culture, which is not aligned with Agile practices:

> *"It doesn't make any sense to me because people game the system. They try to get as many easy stories as they can to make their matrix look really good. It doesn't enforce the whole team spirit."* - P16, Agile Coach.

Performance appraisal solely based on individual assessment is not really applicable for Agile teams. Surprisingly, the findings of this study affirmed the existence of software organizations that shifted to using Agile methods without changing their performance appraisal:

> *"We haven't changed our performance appraisal since we moved to Agile methods. We should do at some point. Our current system is individual performance or behaviour. If we want good team behaviours, I think we have to change."* - P46, Agile Coach.

Therefore, reforming performance appraisal is vital for Agile Software Development projects. Based on the findings of this study, reforming performance appraisal of Agile teams requires two major changes: integrating individual and team performance assessment criteria, and shifting from quantitative to qualitative performance appraisal. The next subsections describe the implementation of these changes in detail.

7.2.1 Integrating Individual and Team Performance Assessment Criteria

In order to align with Agile practices, it is important to integrate team performance appraisal in relation to certain pre-established criteria and

organizational objectives. Team performance appraisals assess the performance of teamwork including an individual's contribution to the team.

> *"We set management [of performance appraisal] by objectives. It is a combination of individual performance and shared goals. We have shared goals and also individual goals."* - P31, Scrum Master.

The findings of this study indicated that the weight of individual and team performance appraisal varies and depends on the organization's goals and objectives. Balancing the measurement of individual and team performance, however, tends to help organizations to address individual skill development as well as focusing on achieving team goals.

> *"We emphasize team performance rather than individual performance."*
> - P32, Scrum Master.

Performance is measured based on the employee's achievements and reflects the significance of the tasks within the organizational goals. Performance relies on the behaviour of individuals in the team. As Agile Software Development projects emphasize effective teamwork by concentrating on people, behaviour is an important indicator of performance appraisal.

> *"In our performance review, we measure two key things. First is performance and second is behaviour."* - P32, Scrum Master.

Behaviour can be appraised on how well Agile team members work with others in maintaining a good social and organizational network. Figure 7.2 shows an excerpt of a performance appraisal that indicates the assessment of behaviour in Agile Software Development teams. The performance appraisal applied at the participant's organization (P32) is specific for Agile Software Development teams.

Good behavioural skills also tend to improve knowledge transfer in Agile teams. Willingness to share knowledge is the key point of success in

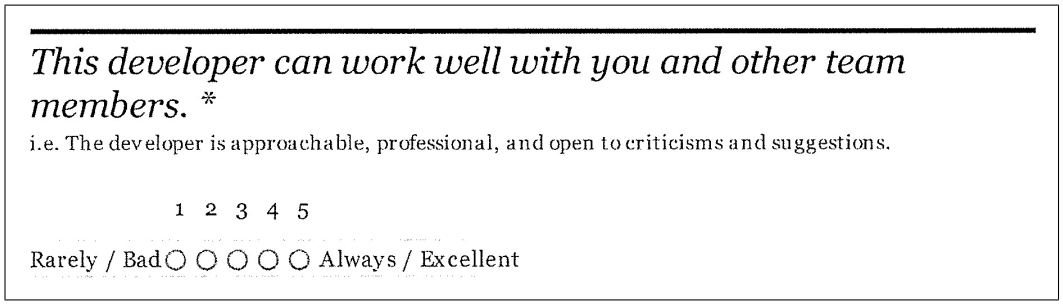


Figure 7.2: Excerpt of Performance Appraisal on Assessment of Behaviour (Provided by Participant P32).

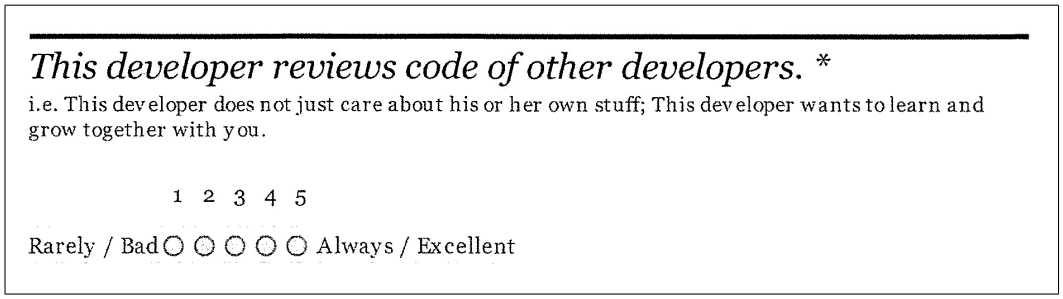


Figure 7.3: Excerpt of Performance Appraisal on Sharing and Transferring Knowledge (Provided by Participant P32).

transferring knowledge among Agile team members. This is clearly shown in the excerpt of performance appraisal depicted in Figure 7.3, which indicates the importance of peer code review as a key point of success in sharing and transferring knowledge.

Expertise coordination relies on the ability of Agile team members to share, preserve and access team knowledge through knowledge transfer (refer to Chapter 5). Knowledge transfer indirectly tends to improve the individual or team skills matrix. Therefore, knowledge transfer criteria should be reinforced in performance appraisal for Agile teams.

> *"We also have communication and knowledge transfer criteria in our performance review. These are important points in our skills matrix. That's why we focus on that."* - P28, Developer.

> *"We have different category skills, such as knowledge sharing, networking, communication skills, and others."* - P28, Developer.

The findings of this study indicated that many possible assessment criteria are used in performance appraisal of Agile teams. The choice of assessment criteria, however, should consider the integration between individual and team performance assessment specific for Agile teams. The selection of assessment criteria should reflect the significance of Agile team members' tasks and responsibilities within the framework of the team's and the organization's objectives.

7.2.2 Shifting to Qualitative Performance Appraisal

A common performance appraisal method is basically based on quantitative measurement, which is represented by using numbers or scores:

> *"The scale is 1 to 5. There is a description for each category and the score. It is like a goal. We can look at what we should improve and what we need to change."* - P28, Developer.

A score enables a superior to indicate the level of their subordinates' achievement or performance, however, it is not adequate to provide feedback on how the subordinates can improve themselves. Thus, it is important to integrate a feedback section into performance appraisal for Agile teams. The score indicates the alignment of employees' performance with the defined key performance indicators (KPIs), whereas the feedback works as an indicator as to what needs to be improved:

> *"The purpose of feedback is for people to improve. The performance appraisal contains feedback."* - P34, Agile Coach.

Feedback can be obtained in a number of ways: observation and peer-review. Observation enables superiors to see and confirm the behaviour

and performance of their subordinates before completing the performance appraisal. Superiors can also identify where and how the subordinates can improve themselves.

The findings of this study indicated that the accuracy of observation relies on how superiors pay attention to subordinates. Superiors should be aware the relationship between subordinates, how they are working together as a team, and their commitment to achieving desired goals.

> *"So, we get a picture through talking to everyone in the team and watching them. Which people are helping the team performance and which people are doing things that hinder the team performance. That's kind of talking to people and observing in action."* - P34, Agile Coach.

The validity of peer-review can be strengthened through observation, which allow superiors to see the consistency between peer-review result and real-situation. Peer-review requires Agile team members to assess their peers. Maintaining confidentiality throughout the peer-review process is vital and only superiors or the management of organization should know the content of the assessment. The assessment should be qualitative by providing verbal or written reviews and comments on how every member works together in the team:

> *"So, my preference is everyone should be giving each other feedback, but qualitative feedback, not number based. [For example] 'you really help me when you do X or you really frustrate me when you do Y.' This information enables someone to act for improvement."* - P34, Agile Coach.

Through our observations at company XYZ, a retrospective meeting was a platform that enabled Agile teams to informally observe and review one another. A retrospective meeting allows the team to reflect on what happened during the current sprint and how to improve the next sprint (see section 2.3.1). Our observations showed the existence of peer-review

during the retrospective meeting, when each team member was required to do the following exercise from the observation notes:

> *The retrospective meeting was attended by a Scrum master, who facilitated the meeting, three developers and one product owner as depicted in Figure 7.4. During the meeting, the Scrum master asked everyone to write constructive feedback on what the other Agile roles need to improve for the next sprint. As developers, they had to give feedback to the product owner on what he needs to improve the sprint. The same thing was done by the product owner to indicate what the developers need to do to improve the next sprint. By using sticky notes, they wrote the constructive feedback in the following format:*
>
> *If I'm a product owner, for the next sprint, I would ..... (from the point of view of the developer)*
>
> or
>
> *If I'm a developer, for the next sprint, I would ..... (from the point of view of the product owner)*
>
> *After five minutes, they placed the written sticky notes on the wall as shown in Figure 7.5. The wall was divided into two columns separating feedback for the product owner and developers. The Scrum master gave an opportunity for everyone to ask questions for clarification on the feedback. Then, everyone was asked by the Scrum master to write actions in order to react towards the given feedback. For instance, the product owner stated that the developers need to deliver consistent documentation even though written by different developers (see Figure 7.6). The developers agreed with the product owners and mentioned that they will use a consistent documentation style and also use GitHub to facilitate the consistency in preparing documents.*

Figure 7.4: A Retrospective Meeting of Team A.

> *Based on the developer's feedback on the product owner's progress, the product owner decided to write more notes in order to ensure the developers understand and track the user stories easily (see Figure 7.7).*

Besides identifying holes or gaps in the current sprint, the retrospective meeting also enabled the team to indicate positive feedback through appreciation, as shown in our notes:

> *The Agile coach asked everyone to express appreciation towards the person who is sitting right next to them. They had to mention positive and supportive feedback, and also accomplishments of that*

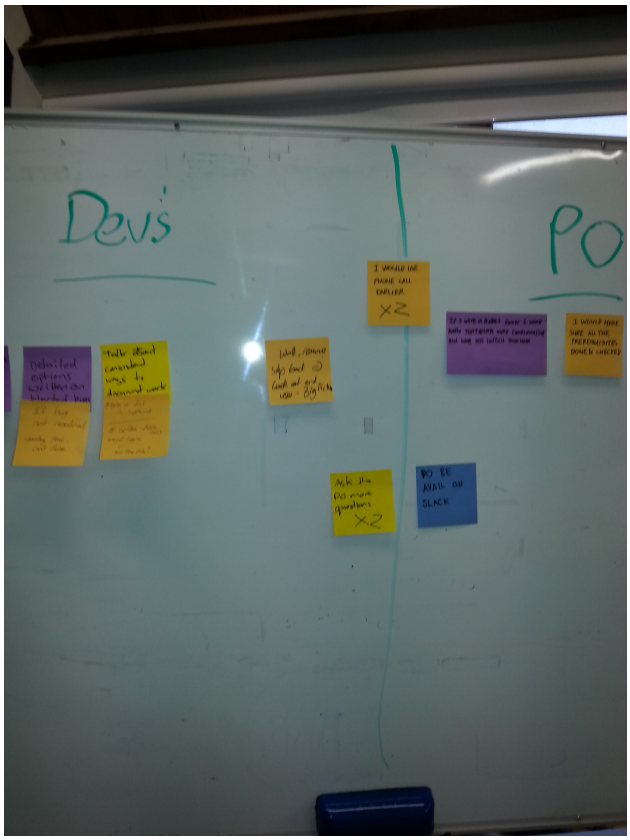


Figure 7.5: Constructive Feedback on the Wall.

> *team member. Everyone had a chance to give positive comments to others and also be evaluated by others. This activity was repeated for each retrospective meeting as a conclusion remark of the retrospective meeting.* [Observation notes]

Although this activity focused more on specific Agile roles rather than individual assessments, it indirectly exposed constructive feedback to the particular individual for improvement. Hence, the team had a space to observe and assess the progress development of their team members' expertise. This information can be used as an input for peer review, which might be relevant to be included in the performance appraisal.

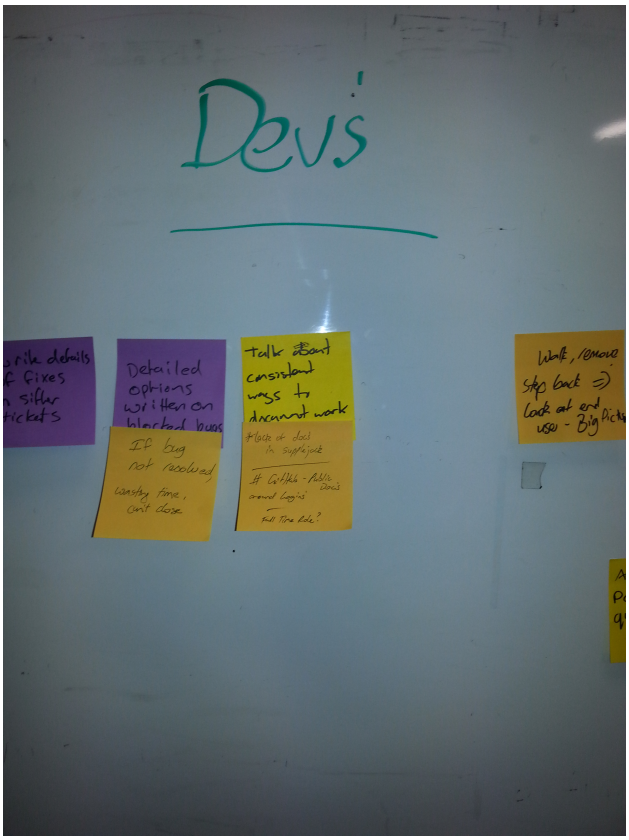


Figure 7.6: Constructive Feedback and Actions for the Developers.

Scoring is not relevant in peer-review because there is a high tendency of Agile teams members to deal with each other in obtaining high scores:

> *"As soon as people are rating other team members, there is a potential of gaming behaviour. For example, 'I will give you a good score if you give me a good score'."* - P34, Agile Coach.

Therefore, peer-review acts as an input for superiors to make decisions on the performance appraisal. Peer-review provides a clear picture of the relationships among subordinates in accomplishing tasks. The final decision, however, is solely from superiors:

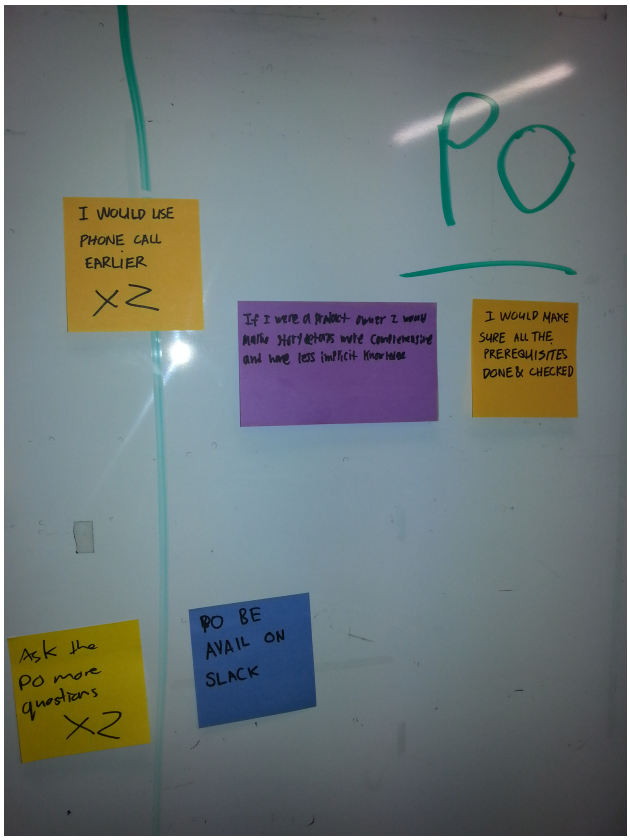


Figure 7.7: Constructive Feedback for the Product Owner.

> *"They may get input from other people, but the decision is still from the management."* - P34, Agile Coach.

After obtaining feedback, a feedback session between a superior and subordinate is vital as an accomplishment on performance appraisal. Participant P34 pointed out that verbal feedback based on point form note is better than providing notes directly to subordinates. The session provides a space for superiors to clearly explain their assessment on subordinates' performance, including their feedback for improvement:

> *"I focus more on conversation. If I write the feedback down, it doesn't give an opportunity for clarification. If I give them the form and they*

> *read by themselves, they might misunderstand what I intended."* - P34, Agile Coach.

Providing scores during assessment of performance hinders the effectiveness of feedback sessions. For instance, Participant P34 claimed that her subordinates tended to focus more on the score rather than listening to the feedback.

> *"As soon as people hear the number, they stop listening to the rest of feedback. When we take the number away, we can focus on the conversation of feedback."* - P34, Agile Coach.

Despite the benefits of qualitative feedback, there is no doubt that scoring systems can be used in performance appraisal of Agile teams. Scoring systems indicate the generic progress development of subordinates and also assist the Human Resources Department in justifying salary increments for Agile team members:

> *"I just use a score to justify the salary. The team has no idea about the score and [the score] couldn't contribute to anything."* - P34, Agile Coach.

A reward system is often related to performance appraisal. We found peer-review is important for rewarding team members through non-monetary compensation such as, prizes and team members' recognition. Participant P45 claimed that her team implemented a gold star chart for representing peer appreciation in her team:

> *"So, when the developer gave me something where I couldn't find errors, I gave him the gold star. Or we have a couple of junior developers and they finished coding on specific things for the first time, so the senior developers gave them the gold star. So you know, it is really cool to do that."* - P45, Software Tester.

As part of self-organizing teams, Agile team members could recognize each other for their help, assistance, expertise, and other significant contributions. This recognition promotes and encourages team members to help each other and indirectly fosters expertise coordination, particularly in sharing expertise:

> *"It is like a peer reward, and motivating each other and recognizing others in the team who have done really good work."* - P45, Software Tester.

> *"So, that's why we set up opportunities like a prize that we called a pony prize. We reward people in terms of expertise."* - P31, Scrum Master.

Even though an Agile team is team-oriented, an individual award is appropriate for acknowledging an individual who has made a significant contribution to the team. As non-monetary compensation, it is impossible for individual rewards to lead to unhealthy competition such as jealousy and resentment. The chosen type of reward, however, depends on the management decision and the acceptance of team members to inspire and motivate them in sharing expertise.

A new direction for performance appraisal of Agile teams will require integration of qualitative feedback in performance assessment. The findings of this study indicated a bidirectional relationship between the expertise coordination process and management support. Feedback tends to support expertise coordination, where Agile teams use the feedback to improve the skills gaps and develop an individual and team's skills matrix. On the other hand, reliable peer-review relies on successful expertise coordination, as inter-dependencies among team members help them to know each other, particularly their expertise area and level.

7.3 Embracing an Expertise Sharing Culture

An expertise sharing culture is vital in coordinating expertise for Agile teams, which requires team members to rely on each other. There is no doubt that every individual should play his or her role in sharing expertise, however, sharing expertise cannot succeed without an expertise sharing culture. The findings of this study revealed three strategies for management support in embracing an expertise sharing culture in Agile teams:

- Fostering a supportive working environment
- Synchronizing goals

7.3.1 Fostering a Supportive Working Environment

A supportive working environment assists Agile team members to perform well and utilize their expertise. Several participants (eg. P2, P5, P13, P14, P31, and P34) suggested that it is very important to identify the constituents of the supportive working environment that tend to boost expertise:

> *"I believe people perform according to the environment. We need to work out what is the environment that can host the people to perform."*
> - P13, Agile Coach.

The findings of this study indicated that two features of working environment tend to boost Agile team members' expertise, as well as coordinate their expertise: open workspace and positive culture. An open workspace without partitions allows Agile team members to work together. It is essential to provide enough space to allow team members to swivel their chairs for ad-hoc conversations or working in pairs:

> *"We have got a very open environment. We didn't have any partitions. There is no cabin. We move around a lot, talk a lot and that's how we work."* - P5, Software Tester.

Open work space facilitates the accessibility of expertise in Agile teams. Team members can easily identify and retrieve the available expertise when it is needed:

> *"It's very important to organize the working environment in the shared area. It is just enough for me to raise my eyes from the keyboard, 'Actually, I'm not sure about this. Can you show me how to do this?' Then, we do it together and fix it together."* - P2, Agile Coach.

> *"All those people come to stand-up [meeting], sitting in the team and are all around me. So, I can hear their conversation. If I don't understand, I can ask them because they are right there. So I think that environment makes it easy for us to learn."* - P45, Software Tester.

On the other hand, separating team members leads to expertise coordination failure. Team members fail to coordinate expertise because they are not aware of what other team members are doing. Participant P2 mentioned that he made a mistake because he failed to access other team members' expertise when facing a problem:

> *"He was too far away and somewhere else. I was going to assume it goes this way. I put this code in this way and that was a mistake because he was not accessible [sic]."* - P2, Agile Coach.

There is a consistency of findings between Participants P2 and P29, which indicated that ease of access tends to reduce the possibility of making mistakes:

> *"When we are sitting next to each other, if we have questions and we can ask immediately. The problem that we don't understand the requirements and tend to doing wrong, is pretty much zero."* - P29, Developer.

The findings of this study also revealed that a supportive environment relies on a positive culture. Agile team members are responsible for cultivating a positive culture through possessing the right mentality and personality. The prevalent mentality in Agile teams is the Agile mind set. It is impossible to create a supportive environment if team members fail to understand and adopt the right Agile values, principles, and practices:

> *"Picking the right mentality, picking the right personality for people in your team."* - P16, Agile Coach.

Positive culture requires Agile team members to possess good personality traits. Participant P14 indicated that the ability to like being with people is an essential personality trait in Agile teams. Relying on other team members through communication, collaboration, coordination, and cooperation requires this personality trait and indirectly fosters a supportive working environment for Agile teams:

> *"I think the Database Administrator or DBA, designer and also the rest of the team want culture. Culture with people. People know what they do, want to help and they are happy to try things out."* - P14, Agile Coach.

> *"The skills that we need to learn, particularly when using Agile [methods] is how do we work with other people and how do we collaborate. So, we can share information and knowledge."* - P34, Agile Coach.

Embracing an expertise sharing culture relies on the ability of Agile teams to work in a supportive working environment. Open workspaces and a positive culture stimulate Agile team members to give the best of themselves to contribute to a common goal. The next subsection discusses how synchronizing goals tend to boost the expertise sharing culture and indirectly facilitates expertise coordination in Agile teams.

7.3.2 Synchronizing Goals

Synchronizing goals means that Agile team members set their team goals together and commit to the goals. Some participants (eg. P3, P6, P11, P16, and P39) asserted the importance of common goals in coordinating expertise:

> *"The best building organization is a sharing and learning organization. Everybody has the same goal and working at the same point [sic]."* - P3, Agile Consultant.

> *"For people to be able to collaborate effectively, they need to have common goals."* - P6, Team Leader.

The management should clearly brief goals that need to be achieved for every team at the very beginning of the software project development:

> *"We run the inception process and the first thing we kicked off was the release planning inception meeting. So, we invited everybody. It is about setting the vision and high level scope [of the project]."* - P16, Agile Coach.

From time to time, Agile teams need to synchronize goals. Agile practices such as sprint planning meetings provide opportunities to synchronize goals among team members. A sprint planning meeting enables team members to define the goals that need to be achieved for the sprint. The sprint goals will lead team members to know what tasks need to be done:

> *"Each person knows exactly what the tasks are and they are really clear about the objectives. To optimize the project, we need clarity about what we are supposed to do next."* - P11, Agile Tester.

Committing to the desired goals requires every team member to rely on each other and to work together as a team. This is the point where

expertise coordination is needed for Agile teams. There is a possibility for team members to depend on each other in sharing and accessing expertise in teams:

> *"To be a team, we must have a common goal. We won't achieve the common goal if we won't work together. For every individual to achieve the goal, we have to help each other. So, each of us is depending on each other."* - P39, Agile Coach.

Commitment to common goals supports expertise coordination in Agile teams. Synchronizing goals enables Agile team members to clearly define what sort of expertise they need in accomplishing the tasks and achieving the determined goals. There is a possibility of identifying and retrieving irrelevant expertise when failing to establish mutual goals.

7.4 Relationships Between Categories of *Management Support*

As depicted in Figure 7.8, *self-selecting teams* and *reforming performance appraisal* are related to *embracing an expertise sharing culture*. The characteristics of *self-selecting teams* and *reforming performance appraisal* are able to support *embracing an expertise sharing culture*.

Knowledge sharing is considered the utmost criteria in *self-selecting teams*. Agile team members choose team members based on who they prefer to share their knowledge with, or who they prefer to learn new knowledge and skills from. Therefore, self-selecting teams tend to support *embracing an expertise sharing culture*.

Embedding knowledge transfer criteria into Agile performance appraisal contributes to the success of *embracing an expertise sharing culture*. In order to meet the knowledge transfer criteria, Agile team members have to share their expertise. This situation indirectly influences *embracing an*

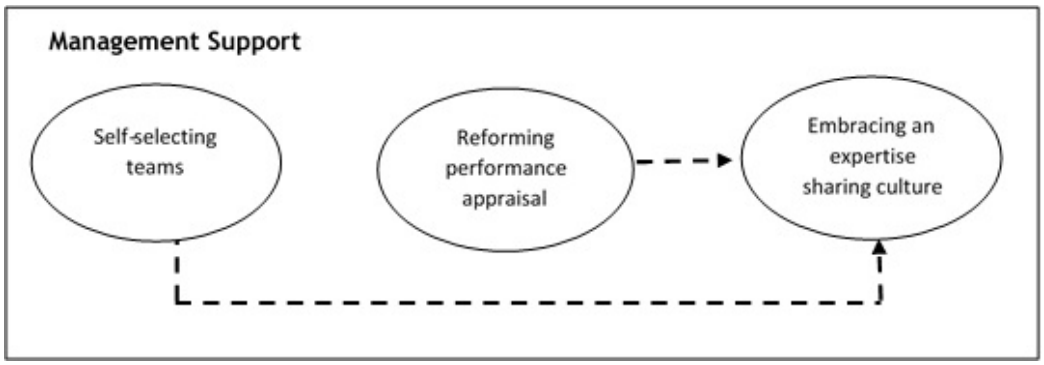


Figure 7.8: Relationships Between Categories of *Management Support*.

expertise sharing culture, which increases team members' willingness to share expertise with others.

Providing qualitative feedback purposely for expertise development also tends to foster *embracing an expertise sharing culture*. Besides self-learning, developing expertise requires Agile team members to rely on each other for extracting expertise. This is the point where sharing expertise is essential for expertise development. Therefore, these characteristics of *reforming performance appraisal* tend to support *embracing an expertise sharing culture*.

Based on relationships between categories of management support, we formulated the following hypotheses:

- H12 : *self-selecting teams will have a positive influence on embracing an expertise sharing culture.*
- H13 : *reforming performance appraisal will have a positive influence on embracing an expertise sharing culture.*

8 Discussion

This chapter discusses the research findings. The first section describes the relationships between categories that led to the emergent theory. The second section provides a detailed explanation of how the results relate to both theoretical foundations and the existing literature. The last section describes how we evaluate and validate the emergent theory.

8.1 Theory Relationships

The emergent theory of this study contains several binary relationships, which indicate the associations between categories as depicted in Figure 8.1. There are three relationships between the core research findings categories:

- A bidirectional relationship between *expertise coordination process* and *management support*
- A unidirectional relationship between *management support* and *coordinating outside expertise*

- A unidirectional relationship between *expertise coordination process* and *coordinating outside expertise*

Successful *expertise coordination process* depends on support from management. On the other hand, *management support* is influenced by *expertise coordination process*. Besides influencing *expertise coordination process*, management also plays an important role in supporting *coordinating outside expertise*. Coordinating expertise outside Agile teams involves *expertise coordination process*.

8.1.1 A Bidirectional Relationship Between *Expertise Coordination Process* and *Management Support*

The relationship between *expertise coordination process* and *management support* is a two-way relationship, where both categories influence each other. Successful expertise coordination relies on the ability of Agile team members to carry out the *expertise coordination process*. Without support from management, however, Agile teams would be unable to coordinate the expertise effectively. At the same time, *management support* also relies on *expertise coordination process*.

Management plays important roles in reinforcing *expertise coordination process* through *self-selecting teams, reforming performance appraisal,* and *embracing an expertise sharing culture.* The findings of this study indicate several features of *self-selecting teams* that facilitate *expertise coordination process*:

- Small teams - A small team with fewer lines of communication and interaction tends to speed up the process of *locating and recognizing expertise,* and *accessing expertise* as well.

- Stable teams - Little fluctuation in team composition enables the stability of skill sets and indirectly expedites *locating and recognizing expertise* and *accessing expertise.*

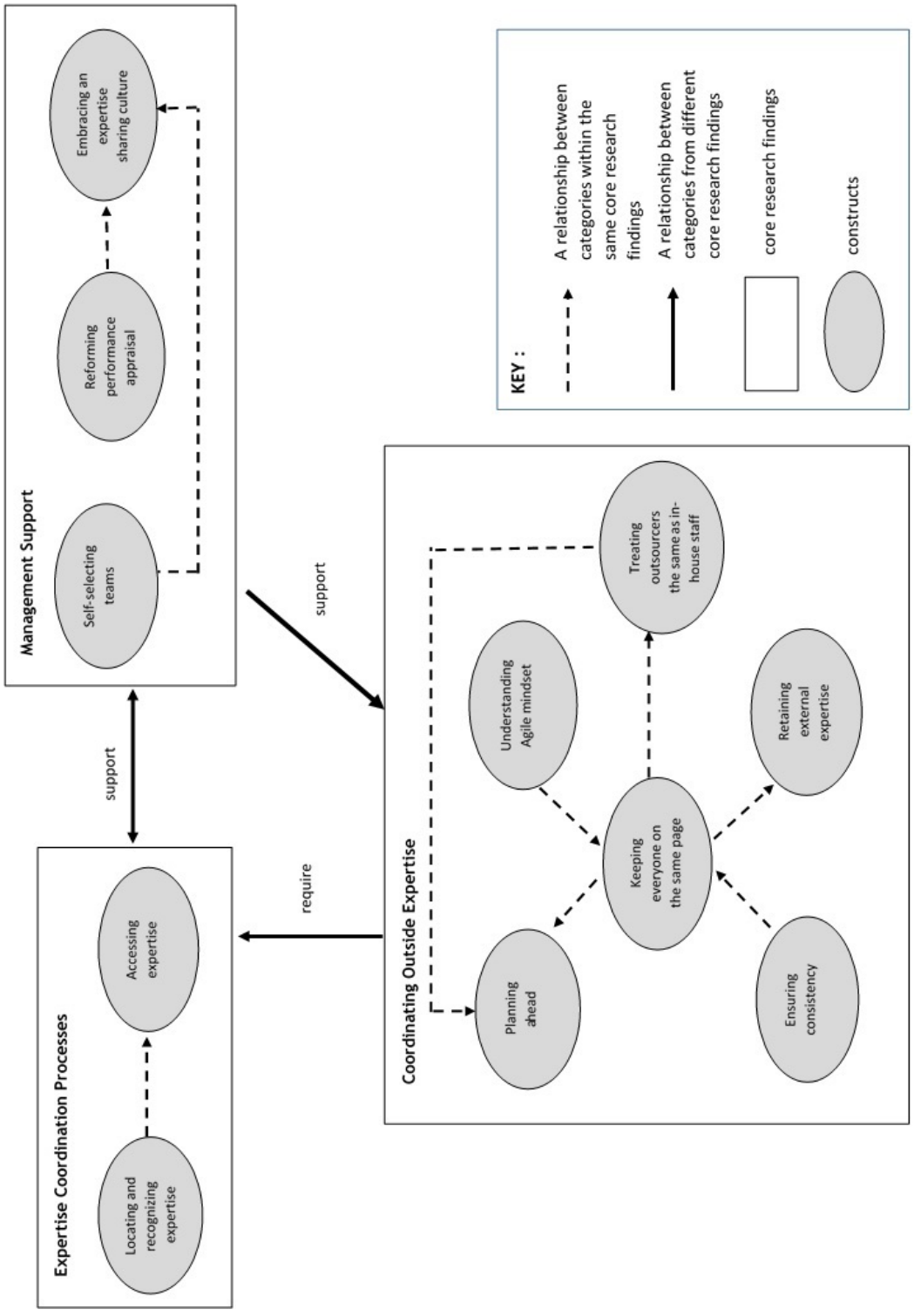


Figure 8.1: The Emergent Theoretical Code of this Study (Reprinted from Figure 4.1.)

Knowledge sharing is a pivotal factor in *self-selecting teams,* where Agile team members put priority on who they can gain new knowledge and skills from, and who they can share and contribute their knowledge with. Therefore, locating expertise is vital to find the right team member with the right knowledge and skills.

Reforming performance appraisal has a relationship with the expertise coordination process through knowledge transfer criteria and qualitative feedback. Knowledge transfer should be considered an essential criteria in performance appraisal of Agile teams. Therefore, the knowledge transfer criteria tends to foster the knowledge sharing culture in Agile teams. In order to ensure successful knowledge transfer, Agile teams tend to rely on each other by identifying and recognizing the owner of expertise, and accessing that expertise.

Providing qualitative feedback instead of scores enables Agile teams to identify room for improvement, especially on the development of expertise. *Expertise coordination process* is able to support the development of expertise, when Agile team members need to rely on each other for extracting a team member's expertise.

Specifically for *embracing an expertise sharing culture*, we found that an open work space, good personality traits, mutual goals, and team recognition are associated with *expertise coordination process*:

- An open work space facilitates the accessibility of expertise in Agile teams.
- Possessing good personality traits enables Agile team members to easily identify the right person with the right expertise and to extract that expertise as well.
- Working towards mutual goals involves identifying and accessing relevant expertise for achieving the desired goals.
- Team recognition relies on the effort that every team member puts to share their knowledge and skills with others.

The roles of management are not restricted to emergent *management support* categories: *self-selecting teams*, *reforming performance appraisal*, and *embracing an expertise sharing culture*. The findings of our study also indicated indirect roles of management in supporting *expertise coordination process*. The success of *locating and recognizing expertise*, and *accessing expertise* depend upon effective support from management, including policies, regulations, and expenditure. For instance, specifically for *locating and recognizing expertise*, the effective usage of an expertise directory relies on how management sets up proper workflows in updating the expertise directory.

Effective self-selecting teams rely on the ability of team members to know each others' expertise area and quantify the level of that expertise. *Locating and recognizing expertise* facilitates the process of *self-selecting teams*, where Agile team members are able to select the right team based on the the team member's capabilities and expertise.

Reliable peer-review through *reforming performance appraisal* relies on inter-dependency among team members to get to know each other, particularly their peers' performance. Knowing peers' expertise area and level is a basis of assessing their peers' performance. Therefore, a relationship between *reforming performance appraisal* and *locating and recognizing expertise* is visible.

On the other hand, *embracing an expertise sharing culture* relates to *locating and recognizing expertise* and *accessing expertise*. Both steps are essential to support expertise sharing and indirectly contribute to successful *embracing an expertise sharing culture*.

Based on the relationships between categories of expertise coordination process and management support, we formulated the following hypotheses:

- H2 : *self-selecting teams will have a positive influence on locating and recognizing expertise*.

- H3 : *self-selecting teams will have a positive influence on accessing expertise.*
- H4 : *reforming performance appraisal will have a positive influence on locating and recognizing expertise.*
- H5 : *reforming performance appraisal will have a positive influence on accessing expertise.*
- H6 : *embracing an expertise sharing culture will have a positive influence on locating and recognizing expertise*.
- H7 : *embracing an expertise sharing culture will have a positive influence on accessing expertise*.
- H8 : *locating and recognizing expertise will have a positive influence on embracing an expertise sharing culture.*
- H9 : *accessing expertise will have a positive influence on embracing an expertise sharing culture.*
- H10 : *locating and recognizing expertise will have a positive influence on reforming performance appraisal.*
- H11 : *locating and recognizing expertise will have a positive influence on self-selecting teams.*

8.1.2 A Relationship Between *Expertise Coordination Process* and *Coordinating Outside Expertise*

Expertise coordination process is involved in different layers of relationships: Agile team members, inter-Agile teams, and Agile team and outside expertise. Interactions and collaborations between Agile teams and outside expertise requires *locating and recognizing expertise.* Before engaging with tasks, both sides need to clearly define who they will deal with, including roles and expertise. It is quite challenging when both sides have to deal

with each other, particularly in a new working environment. Therefore, this study revealed several issues faced by Agile teams and external specialists or outsourcers in *locating and recognizing expertise* as follows:

- A few outsourcers were overconfident in their field of expertise and reluctant to recognize in-house expertise.

- One outsourcer claimed that he preferred working with in-house staff because they knew the available in-house expertise:

Therefore, bridging the gap between Agile teams and outsourcers through *treating outsourcers the same as in-house staff* tends to improve the relationship between them. *Treating outsourcers the same as in-house staff* assists the outsourcers to quickly adapt to an unknown working environment. This situation indirectly requires team members and outsourcers to know each other by *locating and recognizing expertise*.

A relationship between *coordinating outside expertise* and *accessing expertise* exists through *retaining external expertise*. In order to avoid too much dependency on outside expertise, Agile teams put emphasis on retaining outside expertise by applying accessing expertise approaches. Several approaches can be applied to pull outside expertise into Agile teams through *accessing expertise* (see section 5.2).

Based on the relationships between categories of *expertise coordination process* and *coordinating outside expertise*, we formulated the following hypotheses:

- H14 : *locating and recognizing expertise will have a positive influence on treating outsourcers the same as in-house staff.*

- H15 : *accessing expertise will have a positive influence on retaining external expertise*.

8.1.3 A Relationship Between *Coordinating Outside Expertise* and *Management Support*

A relationship between *coordinating outside expertise* and *management support* exists through *ensuring consistency* and *keeping everyone on the same page*. The strategy of *ensuring consistency* ensures stable Agile teams have a low rate of turnover of both team members and external specialists. Considering the stability of the team also relies on how management supports the *self-selecting team.* This is clearly stated in our findings, where *self-selecting teams* assist team stability as team members have an opportunity to choose who they prefer to work with. Therefore, they can remain intact for the long term and contribute to the stability of team.

A strategy of *keeping everyone on the same page* is to ensure effective communication between Agile teams and outside expertise. Conveying mutual goals that need to be achieved at the end of the software development project requires effective communication between Agile teams and outside expertise. Therefore, the successful coordinating outside expertise relies on the role of management in synchronizing goals for *embracing an expertise sharing culture.*

Based on the relationships between categories of *coordinating outside expertise* and *management support*, we formulated the following hypotheses:

- H22 : *self-selecting teams will have a positive influence on ensuring consistency*.
- H23 : *embracing an expertise sharing culture will have a positive influence on keeping everyone on the same page.*

8.2 Expertise Coordination Theory

This section presents a detailed explanation of how the findings of this study relate to both theoretical foundations and the existing literature. The

explanations are presented in the following subsections:

- Expertise Coordination Processes
- Coordinating Outside Expertise
- Management Support

8.2.1 Expertise Coordination Processes

This thesis presents the expertise coordination process that is aligned with expertise coordination as defined by Faraj and Sproull [54]. Expertise coordination requires a team to recognize who has particular expertise, when and where they are needed, and how to access that expertise effectively [54].

The emergent expertise coordination process for Agile teams addresses the first research question of this study: *'how is expertise coordinated in Agile Software Development projects?'* (refer to section 1.1). This thesis describes the expertise coordination process in general and emphasizes how to locate and recognize expertise, as well as access expertise in Agile teams (refer to Chapter 5).

8.2.1.1 Locating and Recognizing Expertise

Locating expertise involves a process of identifying who has which sort of expertise, and what is the level of that expertise (see section 5.1). Faraj and Sproull [54] and Shim et al. [195] assert that knowing the location of expertise is a part of the expertise coordination process in software development projects, as well as recognizing expertise. There is no point having massive expertise available if team members fail to recognize that expertise.

Knowing *"who knows what"* will determine who has particular expertise [54] and indirectly leads to developing meta-knowledge of the available expertise in their team. This is supported by Garrett and her colleagues [62],

who claim that every team member should be aware of others' expertise, particularly the relevant expertise to perform tasks.

In the context of Agile Software Development, Strode et al. [205] posit that *"know who knows what"* will determine who has a particular of expertise in Agile teams. Reasons for locating expertise in Agile teams are depicted in Table 8.1.

Table 8.1: Reasons for Locating Expertise in Agile Teams.

Reason	**Supported by**
to bring the right team member with particular expertise to solve a problem in a timely manner	Lee and Xia [113]
to develop a sense of "*who we are*" and a collective awareness of the available expertise	Bielaczyc and Collins [22] and Strode and Huff [205]
to select tasks by considering the capability of other team members	Hoda et al. [83]
to pair with the right Agile team member, particularly on a new task	Vanhanen and Kopi [217]

There is a need to facilitate identification of knowledge owners or experts in Agile teams [181, 183]. Locating and recognizing the internal expertise in Agile teams, however, is not explicitly reported in the literature. This raises a question: *"How do Agile team members depend on each other in locating expertise in Agile teams?"*. This study revealed six ways Agile teams identify and acknowledge expertise: *communicating frequently*, *working closely together*, *declaring self-identified expertise*, *earning certification*, and *using an expertise directory* (see section 5.1).

Communicating Frequently We found that most Agile team members prefer communication to identify the relevant expertise (refer to section 5.1.1). Dessai and Kamat [49] affirm that team members have opportunities to acquire more detailed information about their peers' expertise through frequent interactions. This is supported by Ehrlich and Chang [52], who posit that frequency of communication positively influences the awareness of team members' expertise.

Working Closely Together The findings of this study revealed that relying on communication is not adequate for locating the right expertise. Communication enables Agile team members to identify the area of expertise, but it is difficult to confirm and quantify the level of expertise. Integrating communication with collaboration will strengthen the process in finding the right expertise in Agile teams. Working closely together enables Agile team members to observe others' expertise directly. Several researchers have revealed that individuals are better at identifying team members' expertise when they spend more time working together [116, 124].

Declaring Self-Identified Expertise Several researchers such as Wegner et al. [227] and Janev et al. [92] suggest that expertise can be identified through self-disclosure. The main issue of self-declared expertise that has been discovered in this thesis is the accuracy and reliability of the expertise. Through communicating frequently and working closely together, Agile team members are able to recognize the self-identified expertise.

Earning Certification Tripp [213] defines a certification as an assessment of an individual's expertise via a standardized measurement instrument. This thesis posits that certification increases the opportunity to identify an individual with knowledge and recognize an individual who has demonstrated a high level of competence in a certain field of expertise (section 5.1.4). Even though there are advantages of earning a certification, this

study affirms that it is not a guarantee of an individual's competency. Reliability of certification, however, relies on the ability of an individual to keep pace with new technologies and software developments, which requires lifelong learning [97, 192, 213].

Using an Expertise Directory Sarma et al. [185] posit that an expertise directory has been proposed as a coordination tool for locating expertise. One surprising point arising from this study is that most Agile team members prefer communication to identify the relevant expertise rather than an expertise directory. Most Agile practitioners were reluctant to use an expertise directory because of the difficulties in keeping the expertise profiles up to date (refer to section 5.1.5). This major drawback aligns with the research finding discovered by Bertoni and Chirumalla [21]. Based on 37 semi-structured interviews conducted at the Swedish Excellence Centre for Functional Product Innovation, they found Human Resources staff faced difficulties in dealing with updating and populating an expertise directory.

Argote and Linda [12] and Bertoni and Chirumalla [21] suggest that through technology improvement such as Web 2.0 tools, the expertise directory can be improved, particularly in updating the expertise profile. Web 2.0 tools provide a communication platform for identifying expertise based on who provides answers to queries, instead of working as expertise directories of declared expertise. This tends to improve the accuracy of the expertise directory.

Leonardi and Treem [116], however, claim that there is a high tendency to fail in capturing and reflecting the real expertise of team members due to inaccurate or incomplete information entered into the expertise directory. As the expertise directory information is visible to everyone, there is a possibility that someone whose position is in jeopardy provides inaccurate expertise information in order to look better than other team members. Even though we did not find any situation of inaccurate information in an

expertise directory, many consequences arise due to inaccurate expertise information. For instance, Leonardi and Treem [116] state that visibility of expertise information assists team members to evaluate and recognize others' expertise. Inaccurate expertise information negatively impacts on the ability to recognize available expertise.

Indirect Ways of Locating Expertise The findings of this study also revealed that Agile values, practices, and principles provide spaces and opportunities for Agile team members to identify and recognize the internal expertise in their teams. For instance, the first two ways of locating expertise in Agile teams: *communicating frequently* and *working closely together* are embedded in the first core Agile value: *"Individuals and interactions over processes and tools"* [59]. Through Agile practices such as stand-up meetings and sprint retrospective meetings, team members also have opportunities to identify the owner of the expertise and quantify the level of expertise.

There are probably other approaches for locating and recognizing expertise in Agile teams which were not discovered in this study. The identified approaches of locating and recognizing expertise, however, can be used as a basis for Agile team members to locate the source of internal expertise and recognize the need for the available expertise.

8.2.1.2 Accessing Expertise

After locating and recognizing expertise, Agile teams can utilize the available expertise by accessing the expertise. Accessing expertise involves a process of retrieving the available expertise and pulling new expertise into Agile teams (see section 5.2). Faraj and Sproull [54] assert the importance of how to access the expertise effectively. The findings of this study revealed five main approaches to accessing expertise: *embracing expert-novice relationships*, *engaging hands-on learning*, *running effective meetings*, *establishing discussion channels*, and *archiving explicit knowledge.*

Embracing Expert-Novice Relationships Our findings identified three basic expert-novice relationships: *Apprenticeship*, *Mentorship*, and *Coaching* (see section 5.2.1). *Coaching* and *Mentorship* are common expert-novice relationships for sharing tacit knowledge in Agile teams [53, 179]. Several research studies such as Lindvall et al. [123] and Misra et al. [146] have demonstrated the success of coaching and mentoring in sharing knowledge in Agile teams. Apprenticeships have also proved successful in creating masters in various fields of expertise [76, 211], however, we found less implementation of apprenticeships in Agile teams. Based on our findings, we found several limitations of apprenticeships which means they are less appropriate in Agile Software Development (see section 5.2.1). As apprenticeships focus on a one-to-one relationship, the spread of expertise is slower than coaching and mentoring. Furthermore, apprenticeships are hierarchical relationships, which are based on authority. This is not aligned with Agile principles in embracing self-organizing teams. Therefore, Ken [93] suggestes that there is a need to figure out the proper implementation of apprenticeships in the Agile Software Development context.

Engaging Hands-on Learning Our findings indicated that Agile methods emphasize hands-on learning for accessing expertise in Agile teams: *coding dojos*, *pair-programming*, *design exercises*, and *internship programmes* (refer to section 5.2.2). A majority of participants asserted that pair-programming was their preference in sharing knowledge among peers and incorporated with other practices such as coding dojos, coaching, and apprenticeship. Based on our findings, User Experience Designers addressed a positive view towards design exercises, which enabled them to coordinate their expertise with customers. Design exercises have been accepted in Agile Software Development projects, since these exercises enable customers to collaborate and contribute in designing their intended software interfaces [56, 135, 164]. Consistent with other studies such as Bannerman et al. [15], Dorairaj [50] and Santos et al. [181] internship programmes enable team

members to travel to another site for working with another team or project in order to support knowledge dissemination. The literature on internship programmes, however, is limited because it focuses more on the academic setting rather than the Agile Software Development context.

Running Effective Meetings The findings from our study revealed the importance of Agile meetings as platforms for assessing available expertise (refer to section 5.2.3). Our findings affirmed Law et al. [109] and Melnik et al. [136], that Agile meetings indirectly facilitate knowledge sharing in teams through communication and collaboration among team members. An interesting point arising from our study is that Scrum of Scrums meetings reinforce expertise coordination, particularly when transferring and sharing expertise occur between Scrum Masters. In conjuction with Agile meetings, our findings revealed that several Agile Software Development projects still run traditional meetings such as meetings for particular roles. Santos et al. [181] and Hussain et al. [89] posit the importance of these meetings in supporting task coordination, as well as inter-team expertise coordination. Inter-team knowledge sharing among the same roles enables synchronizing, synergizing, and balancing particular roles of expertise across each team.

Establishing Discussion Channels The findings from our study affirmed that accessing expertise can be established through discussion channels such as *chatting tools, interest groups, knowledge sharing sessions*, and *robust debate* (see section 5.2.4). Our finding is consistent with Dessai et al.'s study [49], which indicates that chatting tools assist the understanding of others' expertise and also facilitate the sharing of expertise among Agile team members. Similarly to interest groups, our findings align with Verburg et al. [218] and Reychan et al. [175], which consist of members who have a common interest in a particular area of expertise and focus on sharing their expertise. Furthermore, several researchers affirmed adoption of conference

agendas leads to a variety of knowledge sharing sessions in Agile teams, such as open space and lightning talks [138, 181, 182, 196]. Our findings, however, contradict a study completed by Chau et al. [33], which posits that debate tends to delay the knowledge transfer in Agile teams. Fernie et al. [58] argue that the role of debate is an alternative method of knowledge sharing. The effective facilitation of debate is essential to ensure that the knowledge sharing between individuals takes place successfully.

Archiving Explicit Knowledge This thesis describes how explicit knowledge can be disseminated, shared, and preserved in Agile teams through *whiteboards, videos,* and *document management tools* (see section 5.2.5). The usage of whiteboards and videos is common for sharing explicit knowledge in Agile teams. Our findings indicated that most Agile teams use document management tools in sharing their explicit knowledge. As document management tools, wikis gain positive attention among Agile practitioners such as Chau et al. [32] and Garcia et al. [60], serving as knowledge platforms for the Agile Software Development community to organize, share, integrate, and preserve explicit knowledge effectively. Wikis can help in distributing expertise by providing each team member with privilege to access, update, and share their explicit knowledge. Santo et al. [181], however, reveal that wikis fail to work as extensive documents which contain all relevant and needed knowledge. Through a Grounded Theory study, Santo et al.'s study [181] found difficulties in managing the wikis, particularly in capturing knowledge into wikis. This was due to the negative perception towards wikis as knowledge sharing tools, where most participants were reluctant to participate in wikis. In order to energize the importance of document management tools, Settina and Heijstek [201] posit the need of software tools' integration, including wikis, which requires further research on the adoption of these tools.

8.2.2 Coordinating Outside Expertise

This thesis describes how Agile teams manage and utilize outside expertise: external specialists' expertise and outsourced expertise (refer to Chapter 6). The emergent category *coordinating outside expertise* addresses the second research question of this study: *'What are the strategies used in coordinating expertise outside Agile teams?'* (refer to section 1.1). This question attempts to identify strategies used in coordinating expertise outside Agile teams.

This study has provided a positive insight into the strategies Agile teams, outsourcers, and external specialists need to consider when dealing with each other: *planning ahead (S1)*, *understanding Agile mindset (S2)*, *ensuring consistency (S3)*, *retaining external expertise (S4)*, *treating outsourcers the same as in-house staff (S5)*, and *keeping everyone on the same page (S6)* (refer to Chapter 6). These strategies are the basis of managing and utilizing external expertise resources in Agile teams. The next subsections describe how these strategies relate to the literature.

8.2.2.1 *Planning Ahead (S1)*

This thesis presents several reasons of outsourcing for Agile teams (refer to Chapter 6). Availability is a factor that urges Agile teams to outsource external expertise. Outsourcing is a common solution for managing a lack of skills and insufficient staff in a required new technology platform [18, 26, 208]. Cost is also a key driver that impels some organizations to start outsourcing software projects. Sutherland et al. [208] and Ishenko and Oleg [90] report that salary costs for in-house staff are much more than outsourced staff. These results oppose our findings, which are that hiring outsourcers involves more cost than recruiting in-house staff. Most Agile practitioners prefer to be contractors, freelancers, or consultants to earn more income. Consequently, Agile teams have to hire outsourcers even though they cost more than in-house staff.

A shortage of permanent staff for Agile Software Development projects

causes an availability issue of external specialists. Strode et al. [204] reveal that availability is a coordination strategy component in Agile teams. Engaging in multiple projects at the same time causes coordination problems in Agile teams and affects availability. Strode et al. [204] discuss coordination problems in general without specifying expertise coordination. Our findings, however, revealed the implications of external specialists' availability in coordinating expertise in Agile teams.

Our findings confirmed an experience report by Hassan and Elssamadisy [77] addressing the availability issue of DBAs in their software project development. A shortage of DBAs caused failure in accessing their expertise particularly when dealing with database changes. Consequently, waiting for the availability of DBAs caused a bottleneck, and delayed their codes delivery.

In order to deal with the availability of external specialists, our findings revealed the need for *planning ahead*. Fowler et al. [59] discourage planning ahead and emphasize responding to change including team members' expertise: Agile teams should put more priority in trusting in team members' expertise to respond to unpredictable changes rather than planning ahead for a particular team member in doing a specific task. Our study revealed contradictory findings, that dealing with external specialists or outsourcers requires planning ahead for their tasks, including when and how they engage with Agile teams. This is vital to ensure their availability for Agile teams, and enables teams to have a contingency plan if the particular external expertise is unavailable. The plan, however, should be just enough and put together by the Agile teams and the external specialists or outsourcers.

8.2.2.2 *Understanding Agile mindset (S2)*

Understanding Agile mindset (F2) assists outsourcers and external specialists to work with the expectation of Agile teams (see section 6.2). Our findings confirmed a study of User Experience Designers by Kollmann and Sharp [99], where a good understanding of Agile facilitates User Experience

Designers' ability to work in an Agile context. Adopting an Agile mindset led the User Experience Designers to be more flexible and change the way they work to fit with the Agile approach.

Furthermore, our findings indicated several consequences of working with external expertise or outsourcers who are unable to work in the Agile context. Sutherland et al. [209] state that failure to adopt an Agile mindset in outsourcing tends to lead to lower performance of Agile teams.

Our findings indicated the need for understanding Agile methods, without revealing how to educate external specialists or outsourcers about being Agile. Several support mechanisms, however, can be used to educate them about Agile practices, values, and principles. For instance, Roche and Vaquez-McCall [177] point out several methods to educate people in understanding the Agile approach through several techniques: *office hours sessions*, *information radiators*, *on-line discussion forums*, and *Ruck and Maul workshops.*

8.2.2.3 *Ensuring Consistency (S3)*

Our findings indicated that *ensuring consistency (S3)* is needed to cope with frequent turnover of external specialists and outsourcers (see section 6.3). De et al. [46] and Narayanan et al. [153] claim that task reallocation and work disruption occur when there is a high rate of team member turnover. The same thing happens when there is a frequent turnover of external specialists. Our findings indicated that unstable Agile teams hinder external specialists from allocating their expertise to teams. A software process standard assists Agile teams and external specialists to easily adapt to new environments. Furthermore, De et al. [46] state that Agile teams usually expect the new team member or external specialist to adapt to their fast-paced work environment. Despite the external specialist's effort, the Agile teams also need to quickly adapt to the new members or external specialists.

8.2.2.4 *Retaining External Expertise (S4)*

Through this study, we affirmed that *retaining external expertise (S4)* is a crucial strategy of coordinating outside expertise (see section 6.4). This is inline with McConnell's study [134], which addresses transfer of expertise outside teams as a common risk of outsourcing. Therefore, many efforts have been made to retain external expertise in Agile teams. According to Redpath et al. [171], some organizations choose to offer ongoing work or regular positions to outsourcers in order to capitalize on knowledge and experience of outsourcers. Furthermore, Srinivasan et al. [198] claim that some organizations use a strategy of hiring outsourcers who take on junior roles, while their permanent staff take on senior roles. This strategy retains the high-level expertise in Agile teams without letting outsourcers leave the teams with their valuable expertise once the project is finished. As for earning more income, however, our findings indicated that most outsourcers prefer not to accept offers to become permanent staff.

A lack of expertise among Agile team members sometimes urges them to tightly rely on external specialists [77]. So, Hassan and Elssamadisy [77] and Lin et al. [120] state that it is important to ensure external specialists are able to transfer their knowledge and skills to other Agile team members. This is aligned with our findings, which indicated the importance of *knowledge retention (F3)* in developing in-house expertise by transferring and retaining outsourcers' expertise, as well as external specialists' expertise. Moe et al.'s study [150] posits that knowledge retention requires identifying the external specialists' expertise and distributing their expertise. A part of our findings revealed how Agile team members locate and distribute expertise in teams (refer to Chapter 6). These findings are relevant as a basis to locate and retain external expertise in Agile teams, but it is impossible to retain all external expertise in Agile teams. This accords with Moe et al. [150] who claim that there is a need to explore how to define which external expertise needs to be retained in Agile teams.

8.2.2.5 *Treating outsourcers the same as in-house staff (S5)*

Our findings indicated that *treating outsourcers the same as in-house staff (S5)* is a strategy to bridge the gap between outsourcers and Agile teams (see section 6.5). These findings are aligned with Gibbs's [64] study, which found that being excluded from team meetings causes outsourcers to feel unappreciated and to miss out on important information sharing. Consequently, they failed to perform according to the team's expectations.

In order to ensure outsourcers easily adapt to a new working environment, Redpath et al. [171] posit that Agile team members should be provided with outsourcers' information, why the outsourcers are being employed, and how they might assist in achieving the team's goals. Therefore, our findings revealed that frequent contact and engagement tend to provide useful information that assists Agile teams to support the involvement of outsourcers in their teams and offer a hospitable work environment for outsourcers.

8.2.2.6 *Keeping everyone on the same page (S6)*

The findings of this study indicated that conveying sufficient information to outsourcers and external specialists is vital to successful coordinating outside expertise in Agile teams (see section 6.6). There is a consistency between our findings and Hassan et al. [77] and Lin et al. [120], where getting everyone on the same page through frequent engagement of external specialists and outsourcers in Agile meetings will assist conveying sufficient information to outsourcers and external specialists. Likewise, Lalsing et al. [106] indicate that coordinating outside expertise tends to progress once information related to outsourcers and external specialists is distributed effectively to Agile teams. Thus, *keeping everyone on the same page (S6)* depends on the ability of Agile teams and outsourcers to distribute information and knowledge on both sides, as well as between Agile teams and external specialists.

8.2.2.7 Relationship Between Strategies

Keeping everyone on the same page (S6) is a core strategy for coordinating expertise outside Agile teams that relates to other strategies (see section 6.6). It is common to find existing literature that discusses the relationships between these strategies in the context of software development, such as Layman et al. [110], Nuwangi et al. [157] and Martini et al. [132]. There is a paucity of literature, however, which focuses on these strategies in the context of interaction between Agile teams and outsourcers, and external specialists.

The findings of this study indicated a relationship between *planning ahead (S1)* and *treating outsourcers the same as in-house staff (S5)*. Frequent contact and engagement liaise between the both strategies. Through frequent contact and engagement, outsourcers tend to quickly adapt to the unknown working environment and be present in a timely manner when their expertise is needed.

There are possible relationships between the strategies which we did not find in our study. For instance, Melo and his colleagues [46] posit that personality causes team member turnover in Agile teams. As the Agile mindset is very personality driven [30], we classified personality under *understanding Agile mindset (S2)*. However, we did not find a relationship between *understanding Agile mindset (S2)* and stability issues that can be addressed through *ensuring consistency (S3)*.

8.2.3 Management Support

Management support is necessary for coordinating expertise for the following independent strategies:

- Self-Selecting Teams
- Reforming Performance Appraisal
- Embracing Expertise Sharing Culture

According to Bass [17], management needs to take apprioriate actions to develop shared mental models. Therefore, the role of management is vital to support expertise coordination since shared mental models are essential for coordinating expertise. The emergent strategies for management to support expertise coordination address the last research question of this study: *"What are management strategies in supporting expertise coordination for Agile Software Development projects?"* (see section 1.1). This question attempts to determine the roles of management in supporting expertise coordination in Agile Software Development projects. (refet to Chapter 7).

8.2.3.1 Self-Selecting Teams

Self-selecting teams are formed when Agile team members choose their own teams and decide whom they are intended to work with. Even though various different approaches are available to form Agile teams, Olsson et al. [158] and Scott and Pallock [188] affirm that self-selecting teams are effective and suit the Agile Software Development environment. This is aligned with our findings, which indicated that self-selecting teams tend to support expertise coordination by:

- Increasing dependencies between Agile team members
- Ensuring a diverse range of knowledge and skills in teams

We found self-selecting teams increase levels of interaction between team members, which is aligned with Scott and Pallock [188]. Self-selecting teams enable Agile team members to choose teams based on whom they prefer to work with. Good team bonding allows Agile team members to rely on each other in coordinating their expertise resources effectively.

Furthermore, Scott and Pallock [188] also state that a diverse range of expertise can be gained through self-selecting teams. This is aligned with the findings of our study, which indicated that self-selecting teams involve the right mix of expertise and composition number of junior and senior

members in a team. Balancing expertise and levels of experience within teams is vital for successful expertise coordination.

Olsson et al. [158] claim that self-selecting teams are challenging and stressful at the beginning of projects. This is aligned with our findings, which indicated that it is hard for team members to decide whom they will work with and what tasks fit their expertise. Our findings revealed three activities required in facilitating self-selection, which indirectly tend to support expertise coordination (see section 7.1):

- Setting up cross-functional teams
- Selecting the right team
- Balancing teams

Setting up cross-functional teams Successful expertise coordination relies on the ability of self-selecting teams to form cross-functional teams. Cross-functional teams consist of team members who complement each other with different expertise, experience and perspectives [94]. Without diversity of knowledge and skills, there is no point in coordinating expertise in Agile teams. Therefore, management should provide valuable and valid information to set up cross-functional teams during self-selecting teams.

Smaller team formation is essential for cross-functional teams. Melo et al. [137] posit that small teams lead to better communication, alignment, and coordination among team members as well. There is a consistency between our findings and Melo et al. [137], in which minimizing team size tends to support expertise coordination. Table 8.2 summarizes how small teams play important roles in coordinating expertise.

Selecting the right team This thesis points out three factors that need to be considered in selecting the right team: types of work, team members, and knowledge sharing (see section 7.1.2). By comparing these factors, we found that knowledge sharing is a pivotal factor that Agile team members

Table 8.2: Minimizing Team Size for Supporting Expertise Coordination in Agile Teams.

Small team characteristics	**Roles in expertise coordination**
Fewer communication lines	Identifying and sharing expertise easily
Tightly bonded relationship	Inducing awareness of team members' expertise
Accessible	Accessing relevant expertise in a timely manner

consider when selecting their team. Therefore, the self-selected teams should consist of a mixture of expertise. Juli [94] posits that having team members with different backgrounds and expertise gives them opportunities to learn and view things from different perspectives.

Our findings, however, contradict Russell and Goodnight's study [178], which claim that self-selecting teams do not tend to have members of divergent expertise, due to a tendency of team members to choose members who work in close proximity. This is opposed to the findings of our study, which revealed that Agile team members put a priority on knowledge sharing and choose team members who possess knowledge and skills that they can gain from and contribute to enhance the team knowledge. Team member factors such as working well together are also considered for successful knowledge sharing, but these are not restricted to members who work in close proximity.

Even though the empowerment of self-selecting teams belongs to team members, our findings posited that facilitation from coaches and management of organization are still needed in forming teams. These findings are supported by Scott and Pollock [188], who claim that the facilitation process during team formation is valuable and leads to an efficient process. Guidance from coaches and management assist Agile teams to form the right team with the right balance of expertise.

Balancing Teams This thesis presented the need to balance teams during team formation (see section 7.1.3). Balancing teams tends to improve team composition with the right roles and number of team members, as well as well-balanced levels of experience within teams. Our findings are consistent with Cohn [40], who posits that team composition should reach the right balance of expertise, experience, and diversity of team members.

Balancing teams also leads to good team stability. Middleton and Joyce [139] claim that a stable team with low staff turnover is prevalent in Agile Software Development projects. As depicted in Table 8.3, our findings align with previous research such as Melo et al. [137] and Middleton and Joyce's [139] study on how a stable team supports expertise coordination in Agile teams.

Table 8.3: Forming Stable Teams for Supporting Expertise Coordination in Agile Teams.

Stable team characteristics	**Roles in expertise coordination**
Stable skill set	Identify team members' expertise easily
Tightly bonded relationship	Increase the team members' sense of responsibilities and commitments toward team performance
Minimal staff turnover	Avoid losing critical knowledge and skills when an expert member leaves the team

A stable team also tends to affect customer engagement in Agile Software Development projects. According to Narayan et al. [154], reallocation of tasks and work disruption occur when there is high team member turnover. This situation tends to affect the understanding of requirements and development of mutual trust between customers and Agile teams. The expertise dependencies between customers and Agile teams are important

to ensure the setting of correct expectations of the software product. The perspective of customers' engagement for coordinating expertise, however, is not covered in this thesis. This is one of the limitations of this study and is described thoroughly in the next chapter.

8.2.3.2 Reforming Performance Appraisal

This thesis points out the necessity of performance appraisal in Agile Software Development projects for:

- Identifying gaps in team member's skills and competencies.
- Enabling expertise development, which involves breadth of skills and specialization in a few areas of expertise.

Noori et al. [156] claim that good performance appraisal enables team members to clearly understand Key Performance Indicators (KPIs) that need to be achieved. Based on KPIs, team members are able to identify their strengths and capabilities, and their lack of expertise. This enables management to take the right steps in dealing with the lack of expertise. This is the point where the expertise coordination process is essential for the development of expertise in order to fill the expertise gaps.

Based on our findings, however, some Agile Software Development organizations still apply traditional performance appraisal solely based on individual assessment. According to Alnaji and Salameh [8] and Coyle et al. [43], it is vital to have an Agile-compliant performance appraisal, as Agile Software Development is team-oriented. An Agile-compliant performance appraisal should be aligned to the Agile values, principles, and practices, which advocate interaction, collaboration, teamwork, and knowledge transfer.

Through this study, we found two major changes for reforming performance appraisal: integrating an individual and team performance assessment criteria, and shifting from quantitative to qualitative measurement of performance appraisal.

Integrating an Individual and Team Performance Assessment Criteria The initial step in constructing performance appraisal is the determination of performance criteria. Besides individual assessment criteria, our findings indicated that team performance criteria should be included in Agile-compliant performance appraisal, to reflect the team-level performance. This is aligned with Alnaji and Salameh's study [8], which posits the importance of individual and team performance assessment criteria in performance appraisal, in order to fit into an Agile Software Development environment.

Our findings support Noori et al.'s study [156] which indicates the need for selecting and updating performance criteria. The performance criteria should be aligned with Agile values, principles, and practices, as well as an organization's goals and objectives. Furthermore, balancing the right weight for individual and team performance assessment criteria is vital to reflect the true abilities of individual and Agile teams as well. Developing team-based performance criteria, however, should reflect Agile core values, principles and practices.

Although there are various team performance assessment criteria, this thesis discusses only two assessment criteria in detail: behaviour and knowledge transfer (see section 7.2.1). These criteria emerged from the findings of this study and are essential for assessing expertise in Agile teams.

This study indicated the importance of evaluating behavioural skills that reflect the abilities of Agile team members in coordinating expertise. Coyle et al. [43] and Alnaji and Salameh [8] claim that the criteria for performance appraisal often focuses on technical skills, whereas Agile Software Development requires a greater emphasis on behavioural skills. According to Shakir [190] and Jackling and Sullivan [91], behavioural skills (also known as soft skills or people skills) include communication skills, conflict resolution, personal effectiveness, creative problem solving, and team building. As expertise coordination involves inter-dependencies

among team members, Agile team members should demonstrate good behavioural skills in coordinating expertise. Balancing technical and behavioural skills is essential in Agile performance appraisal.

Our findings revealed the need to reinforce knowledge transfer criteria in performance appraisal. Noori et al. [156] state that good behavioural skills also lead to an increase in the ability to transfer knowledge in Agile teams. Knowledge transfer facilitates the process of generating a *T-shaped person* [24, 95] in Agile team members, who have breadth in a number of areas, and depth in a few areas of expertise (see section 2.3.3). Knowledge transfer requires the expertise coordination process for identifying and recognizing the needed expertise, and retrieving the expertise.

Shifting from Quantitative to Qualitative Measurement of Performance Appraisal This study indicated the importance of qualitative feedback in Agile performance appraisal (see section 7.2.2). Tripp and Riemenschneider [212] also indicate the importance of qualitative feedback in providing the employee with clear information about his or her performance. Qualitative feedback enables Agile team members to identify areas of strength and weakness, analyse their performance gaps, and take action for improvement. Our findings are aligned with Ahmad and Bujang's study [4], which posits that qualitative feedback does not replace the scoring appraisal, but it can be used as an additional method of appraisal.

Through our observations, feedback can also be gained during retrospective meetings. These meetings allow Agile teams to reflect on the work process used and how to improve the process for the next iteration. The outcome of retrospective meetings is feedback, which tends to influence the performance appraisal. Our findings, however, contradicted Shankarmani et al.'s study [191], which claimes that Agile teams do not need performance appraisal other than retrospective meetings. Based on our observation, relying on the retrospective meetings only is not adequate for gaining feedback and ensuring the reliability of performance assessment.

Therefore, this study indicated that peer-review can be used to gain reliable feedback for performance appraisal. Based on Coyle et al. [43], 360-degree feedback appraisal involves peer-review, where team members can act as evaluators, as well as being evaluated. A reliable peer-review relies on good team bonding. Peer-review indirectly tends to support expertise coordination in Agile teams, which requires inter-dependencies among team members for getting to know each other, particularly their expertise area and level.

Furthermore, feedback also tends to influence successful expertise coordination. As feedback identifies performance gaps and actions for improvement, Agile teams need to coordinate expertise in order to improve or develop an individual and team's expertise. Through the expertise coordination process, Agile team members are able to locate, recognize, and retrieve available expertise for improving an individual and team's expertise.

Our findings revealed that Agile teams implement peer-to-peer rewards which focus on individuals. This contradicts several studies, which affirm that team rewards are more appropriate for Agile teams rather than individual rewards since Agile Software Development emphasizes cooperation between team members [47, 107, 155]. Lapham [107], however, states that Agile teams often choose to reward someone based on individual recognition. Besides individual achievement, peer-to-peer rewards recognize peer contributions including expertise contribution. Several researchers have shown that these individual rewards can promote knowledge sharing, when team members contribute their expertise for helping each other [105, 125]. This indirectly supports expertise coordination, when sharing expertise requires team members to rely on each other for locating, recognizing, and accessing expertise.

Chow et al. [35] affirm that a reward system is a success factor for Agile Software Development. Denning [47] and Vinekar et al. [220], however, suggest the need for a reward system that is suitably designed

for successful adoption of Agile methods. Implementing a suitable reward system tends to support expertise coordination. A proper balance between individual and team rewards is essential for successful expertise sharing. Non-monetary compensation such as training opportunities and special recognition supply motivation to embrace the knowledge sharing culture in Agile teams and indirectly influence successful expertise coordination.

8.2.3.3 Embracing an Expertise Sharing Culture

Management play important roles in embracing an expertise sharing culture in Agile teams (see section 7.3). An expertise sharing culture is the predominant strategy for coordinating expertise in Agile teams through:

- Fostering a supportive working environment
- Synchronizing goals

Fostering a Supportive Working Environment This study indicated the importance of a supportive working environment in facilitating and supporting expertise coordination. This thesis discusses two features of a supportive working environment supporting expertise coordination: open workspaces and positive culture (see section 7.3.1).

Several studies found open workspaces appear to support coordination, including expertise dependencies [34, 137, 145, 223]. These studies show that open workspaces facilitate communication and enable team members to exchange knowledge effectively. In line with the findings of this study, open workspaces allow team members to work closely with each other and be easily accessible. Indirectly, this tends to increase team members' awareness of each other and support expertise development.

We found *reforming performance appraisal* tends to support *embracing an expertise sharing culture*. In contrast, Wang et al. [222] found a bidirectional interconnection between both categories through open workspaces, which did not appear in our findings. They reveal that open workspaces also

contribute to successful performance appraisal because Agile teams can conduct peer-review easily through observing other team members.

According to Cho [34] and Vischer [221], open working environments lead to a high possibility of interruption and disruption in Agile teams, that prevent team members from concentrating on their tasks. Mishra et al. [145] suggest the need for non-verbal communication and behaviour cues, such as posture and eye gaze, in order to reduce interruption and distraction. For instance, our findings revealed that team members indicate their availability for interaction and communication when they look up and make eye contact with others.

Mishra et al. [145] and Omar et al. [159] indicate that cultivating a positive culture in the Agile working environment through an Agile mind set and positive personality is essential for Agile teams. This is aligned with our findings that positive culture tends to support expertise coordination in Agile teams. Possessing positive personality traits such as communication, consideration, and helpfulness, tends to boost good relationships among team members in managing their expertise resources effectively. For instance, Omar et al. [159] found that a positive personality influences the ability of Agile team members to share expertise, particularly in solving problems. Besides individual efforts, management should play important roles in developing positive personalities by establishing people-skills development programmes such as mentor-mentee programmes and team building activities.

Synchronizing goals Through this study, a common goal has been defined as a supporting element for embracing an expertise sharing culture in Agile teams and indirectly tends to support expertise coordination (see section 7.3.2). In line with the coordination definition (see section 2.1), working towards a common goal is the objective of coordination and establishes commitment during the coordination process. This is supported by Strode and Huff, [205] who affirm that a common goal is a key component

of coordination in Agile teams.

This thesis points out that synchronizing goals is beneficial to coordinating expertise in Agile teams. Common goals lead Agile teams to work in the same direction by agreed upon roles and expertise involved. At this point, mutual goals assist team members to identify and retrieve the relevant expertise in Agile teams. This indirectly increases Agile team members' awareness of expertise needed and willingness to fill expertise gaps for achieving the common goals.

8.3 Evaluating the Theory

Numerous approaches exist for evaluating the quality of a qualitative study as a whole, as well as for specific research methods. According to Creswell [44], it is up to researchers to determine which evaluation approaches suit their study. He suggests that researchers employ multiple evaluation approaches to improve the accuracy of their studies. The evaluation of a theory should be looking for different angles such as paradigms and type of research, without being strictly limited to a specific research method [112]. Therefore, this study employed several evaluation approaches for evaluating the emergent theory. In particular, this study employed Lincoln and Guba's approach [121] for evaluating the emergent theory as a part of the qualitative study. This approach focuses on reliability, validity, and credibility of the data. In order to align with Glaser's approach, we used Glaser's evaluation criteria instead of Strauss and Corbin's approach. Glaser's evaluation approach focuses on evaluating the adequacy of research processes and the grounding of the research findings. In order to evaluate the quality of the theory, this study applied Weber's approach to pinpoint its strengths and weaknesses. A detailed explanation for each evaluation approach is described in the following subsections.

8.3.1 Lincoln and Guba's Criteria

Lincoln and Guba [121] define trustworthiness criteria for evaluating qualitative research: credibility, transferability, dependability, and confirmability. These criteria parallel the criteria of internal validity, external validity, reliability, and objectivity, which are widely accepted and used in qualitative research and lead us to apply Lincoln and Guba's approach [44]. This evaluation approach is also aligned with our interpretivist research paradigm. The following subsections discuss thoroughly how each criterion is considered in our study.

8.3.1.1 Credibility

Lincoln and Guba [121] refer to credibility as the truth value of findings. According to Mile and Huberman [140], credibility can be drawn from the following questions:

- "Do the findings make sense?"
- "Are they credible to people we study and to our readers?"
- "Do we have an authentic portrait of what we were looking at?"

In order to evaluate credibility, Guba and Lincoln [121] propose six techniques as follows: prolonged engagement, persistent observation, triangulation of data, peer debriefing, negative case analysis, and member checks.

Prolonged Engagement Prolonged engagement requires researchers to determine whether they spent enough time on the research site and had intensive contact with participants [121]. Our study was conducted over two years and involved 48 participants, with approximately 45 minutes to one hour of semi-structured interviews for every participant. Observations have also been incorporated in our study, which enable us to gain a

deeper understanding of the participants' settings. Therefore, we strongly believe that our study is lengthy and has intensive contact with participants involved.

Persistent Observation Persistent observation addresses whether researchers have conducted an in-depth study to gain detailed explanations of important findings [121]. Moreover, persistent observation should lead us to determine whether we have gathered enough details. Through constant comparison and theoretical sampling in Grounded Theory, we ensured persistent observation in this study. Constant comparison involves continuous interplay between data collection and data analysis (see section 3.4.3), which indicates similarities and differences between emergent codes. We collected the next data based on the existing findings through theoretical sampling. Theoretical sampling helped us to ensure comprehensive findings by gathering new insights or refining and expanding existing findings. We continued to collect data until theoretical saturation had been reached. Constant comparison and theoretical sampling provided integrated approaches for ensuring the credibility of our study.

Triangulation of Data Triangulation of data allows researchers to determine convergence among multiple or different sources of methods and data. Our study includes two observations in conjunction with the 48 interviews and document analysis for strengthening findings through different data sources (refer to section 3.4.2). This study also emphasizes different data sources by ensuring a wide variety of participants with different roles from different organizations based in different countries.

Peer Debriefing Peer debriefing provides ongoing discussion for external review of the research process. Peer debriefing for our study, however, was limited to academic supervisors for ensuring the accuracy of Grounded Theory processes.

Negative Case Analysis Negative case analysis is purposely to identify findings that differ from researchers' expectations, assumptions, or working theories. Through constant comparison, we continuously compared new data with existing findings and sometimes we discovered a few negative cases, which required us to make modifications to the emergent theory.

For instance, we found an opposite perception towards the usage of expertise directories for locating where expertise resides. A few participants, however, claimed that communication is more valuable when finding relevant expertise than using an expertise directory. This finding was considered a negative case, which contradicted the existing findings and led to unexpected outcomes. Therefore, it is important to consider this negative case to draw a firmer conclusion.

Member Checks Member checks require participants' input for confirming data and interpretation. Researchers need to take data, analyses, interpretation, and conclusions back to participants for evaluating the credibility criteria. According to Lazenbatt and Elliott [112], however, a Grounded Theory study does not require researchers to return to participants for checking whether the participants agree with the researchers' interpretation of data. We returned to participants, however, for clarifying ambiguous, unclear, and incomplete data transcripts. If clarification was needed, we would email probing questions to our participants related to clarifying and confirming some data and interpretation. Furthermore, we also emailed our published conference papers to participants for obtaining their feedback. These activities are essential to establish correctness of data interpretation and strengthen the findings of this study. The feedback from participants as follows:

> *"This paper is a reflection of the initial states and challenges the team is looking to and have just begun their transition to Agile methods of collaboration and value creation. Misconceptions of Agile methods as enabling high-performance and being a 'silver bullet' to resolve*

common issues in complex projects frequently results in many of the examples of challenges as highlighted in this paper. Creating sustained behavioural changes to unlock up-sized productivity improvements is truly challenging and this paper has found many common barriers to that end [sic]." - P31, Scrum Master.

"I read your study and I think it is excellent. Good Job! Your study reflects the cordination and collaboration with specialists or teams outside of an Agile team." - P33, Project Manager.

8.3.1.2 Transferability

Transferability refers to the degree to which the research findings can be applied in other contexts [121]. Mile and Huberman [140] list relevant questions for determining transferability criteria. For example:

- "Do the findings include enough *thick description* for readers to assess the potential transferability and appropriateness for their own settings?"
- "Is the sampling theoretically diverse enough to encourage broader applicability?"
- "Does the researcher define the scope and the boundaries of reasonable generalization from the study?"
- "Does the report suggest settings where the findings could be tested?"

In order to answer the above questions, Guba and Lincoln [121] propose thick description of data for evaluating transferability criteria. Thick description of data requires us to provide dense and detailed explanations of concepts, categories, and the theory itself. 48 participants in this study allowed for rich description of findings. We also provided a detailed description of the participants (see section 3.4.2) and boundaries of this

study (see section 4.6) in order to achieve transferability. This enables readers to make decisions about the applicability of our findings in other contexts of the study.

Transferability relies on the sampling used in this study. Through Grounded Theory, theoretical sampling was used to determine the future data based on the current data analysis of this study. This is important to ensure the comprehensive nature of data drawn from a broad range of participants.

The findings of this study, however, do not require further testing because the purpose of the Grounded Theory is to discover a theory of the phenomena studied, but not to test to the emergent theory [67].

8.3.1.3 Dependability

Dependability allows researchers to determine whether the research process is consistent and leads to the stability of the data over time [73, 121, 140]. Mile and Huberman [140] list relevant questions for determining dependability criteria. For example:

- "Are the research questions clear, and are the features of the study design congruent with them?"
- "Do findings show meaningful parallelism across data sources (informants, contexts, times)?"

In this study, we formed research questions (see section 1.1) in general, but within a specific topic of study. The research questions evolved over time depending on the emergent findings of this study. In order to answer the research questions, we had proper research processes (see section 3.4). Therefore, detailed and comprehensive documentation of the research processes has been produced in this study.

Guba and Lincoln [121] recommend researchers use an external auditor to examine the research process and artefacts. In this study, the review

process was carried by academic supervisors rather than an external auditor. The academic supervisors worked together to ensure the findings of this study emerged from interview quotes and observation notes.

8.3.1.4 Confirmability

Confirmability criteria is related to bias issues and the prejudices of the researchers [140]. Confirmability ensures data, interpretations, and findings are rooted in the context of study and participants, apart from the researcher's assumptions or imagination [73]. Mile and Huberman [140] list relevant questions for determining confirmability criteria. For example:

- "Are the study's general methods and procedures described explicitly and in detail?"
- "Can we follow the actual sequence of how data were collected, processed, condensed/transformed, and displayed for specific conclusion drawing?"
- "Are conclusions explicitly linked with exhibits of condensed/displayed data?"
- "Has the researcher been explicit and as self-aware as possible about personal assumptions, values and biases, affective states - and how they may have come into play during the study?"

Guba and Lincoln [121] posit the need for an audit trail in establishing confirmability. An audit trail is included in Chapter 3, which presents examples of research artefacts such as data transcripts, coding structure, and memos. This evidence is described explicitly through Grounded Theory procedures. In Chapters 5 to 7, we included examples of direct quotes from the data to indicate links between data and findings. The confidentially and anonymity requirements, however, prevent us from providing more details of evidence in this thesis.

In term of researcher bias, we were aware of our responsibility to avoid personal assumptions and biases in this study. We attempted to put aside personal perspectives and assumptions in understanding our participants' interests and point of view. Furthermore, researcher bias can be mitigated through memoing, which controls distortion during data analysis [112].

8.3.2 Glaser's criteria

We also evaluated our research according to Glaser's criteria: fit, work, relevance, and modifiability [66, 67]. These criteria are related to each other and fit forms the basis for the other criteria [126]. The detailed explanation for each criterion is described in the next subsection.

8.3.2.1 Fit

Fit refers to the ability of theory to emerge naturally from data without forcing the data to fit preconceived ideas [71]. The emergent categories and their properties should fit the realities of participants' views and phenomena studied. Therefore, maintaining an open mind without any pre-existing theoretical expectation is a fundamental principle of Grounded Theory [67]. In this study, we established fit through a minor literature review and theoretical sampling.

Conducting a minor literature review at the beginning of this study was to ensure the emergent concepts and categories were fit without imposing any preconceived ideas from the literature. The minor literature review led us to determine the area of study and to formulate research questions. Then, we gathered relevant data and identified the real concerns of participants without being influenced by ideas from literature.

According to Urguhart et al. [215], theoretical sampling is a contributor to the fit criteria, in which the emergent theory is truly grounded in data. Theoretical sampling guided us in determining participants based on the existing research findings. This process was continued until it reached

theoretical saturation; therefore, the possibility of imposing preconceived ideas on the data was reduced.

8.3.2.2 Work

Work means the theory should explain, interpret, and predict what is going on within the context of study [66]. The emergent theory should represent the participants' concerns and demonstrate the application value [71]. In order to determine work criteria, we used the acceptance of the theory published and represented in several well-established Agile conferences as a benchmark. The findings of this study were published and presented at eXtreme Programming conferences (XP2014 and XP2015) and the Agile conference in the United States. The papers were well received by conference attendees, as well as paper reviewers, as follows:

> *"The paper tries to theorize the role of external actors in Agile Software Development that is relatively novel in different ways. First, the paper tries to build theory that explains how 'Agile works' - which is rarely found in current literature. There is for sure too little focus on the complexity of these aspects in the current literature, and also too little awareness among practitioners. Hence, although the paper is theoretically framed, the paper also provides practitioners with important insights."*

> *"The topic of the paper potentially fills a gap in current research on Agile Software Development. Apart from the few studies mentioned in the paper, there are to my knowledge no studies that shed light on this particular important issue. Secondly, the paper presents truly rich empirical material that gives the reader in-depth understanding of some of the many complexities that are involved in communicating and coordinating with external experts. Interviews with 48 informants are impressive, and should give a sound basis for theorizing."*

8.3.2.3 Relevance

Relevance refers to the ability of the emergent theory to be related to core problems and processes in the area under study [66]. One way to find out the relevance of the emergent theory was discussing emerging conceptual categories with academic supervisors. Furthermore, we also gained well-received feedback from paper evaluators on the relevance criteria:

> *" This reviewer finds it very interesting that this theorizing emphasizes the relationships between external actors and the Agile team."*

8.3.2.4 Modifiability

Glaser [68] defines modifiability as "*the quality of the theory to be ready for changes to include variations in emergent properties and categories caused by new data*". The emergent theory should be flexible to adapt to modifications in the future [66]. Through constant comparison, the concepts and categories continuously evolved and led us to modify the emergent theory. For instance, the findings of this study revealed numerous unexpected issues in coordinating expertise outside Agile teams. The emergent theory has been modified in order to present these findings and indirectly led us to modify our research questions. This is supported by Charmaz [31], who claims that researchers tend to modify their research questions once they have discovered significant new findings.

8.3.3 Weber's Criteria

In addition to Glaser's criteria, this study has evaluated the quality of emergent theory based on Weber's five criteria: importance, novelty, parsimony, level, and falsifiability [225].

The importance of a theory is assessed based on the importance of its focal phenomena from the viewpoint of practice and research [225]. Our theory addresses expertise coordination in Agile teams, which is deemed

important from the viewpoint of practice. The theory embodies specific processes and strategies that are related to expertise coordination in Agile teams. Therefore, this thesis provides an overview of how Agile teams are able to locate and assess the available expertise, and outside expertise as well. This thesis also presents how Agile teams, external specialists, and outsourcers depend on each other to manage and utilize outside expertise. This thesis also discusses several management strategies to support successful expertise coordination in Agile teams. This study provides Agile practitioners with important insights into expertise coordination in Agile Software Development projects.

From a research perspective, this study discovered there is a paucity of studies that focus on expertise coordination, particularly empirical studies. The findings of this study are important contributions to the body of knowledge in Agile Software Development, specifically coordinating outside expertise Agile teams. Furthermore, the emergent theory can serve as working hypotheses that could be tested by future researchers.

The novelty of a theory relies on the value of the research and its contributions to the body of knowledge [225]. Weber [225] suggests a theory is novel if its focal phenomena has not been covered by existing theories. Our emergent theory is novel since this study focuses on expertise coordination in Agile teams, which has a limited number of previous studies, particularly empirical studies. This study embodies several further investigations that have been proposed elsewhere, such as:

- to facilitate finding knowledge owners or experts in Agile teams [184].
- to explore the interaction between Agile teams and roles outside the teams in coordinating expertise [194].

A simple theory with a small number of categories and relationships is deemed to be parsimonious [225]. We argue that our study has produced a parsimonious theory since the emergent theory consists of six categories: *locating and recognizing expertise, accessing expertise, coordinating outside ex-*

pertise, self-selecting teams, reforming performance appraisal, and *embracing an expertise sharing culture.* The number of our categories is no more than seven, which is in line with the 'small number' suggested by Miller [141]. The parsimonious theory, however, should capture the richness of our focal phenomena, which is described in detail in Chapters 4 to 6.

For evaluating the quality of a theory, Weber [225] suggests researchers determine the level of theory: *micro-level*, *middle-range level*, or *macro-level*. Our study has produced a *middle-range level* theory, which should model neither too narrow nor too broad a range of phenomena. This is in line with Glaser's recommendation to build *middle-range* theory in Grounded Theory study [31]. Our middle-range substantive theory explains how to coordinate expertise, specific to Agile Software Development projects.

The last criteria is falsifiability, in which a high-quality theory could be falsified via empirical tests [225]. Researchers can test the theory under a variety of conditions. For instance, the emergent theory of this study could be tested under the implementation of expertise sharing in a large healthcare organization, which involves interactions and inter-dependency among healthcare practitioners, such as specialists, doctors, nurses, and dieticians.

For this study, however, the emergent theory does not need further testing. This is in line with Glaser [67], who claims that Grounded Theory study emphasizes an inductive approach and emerges from data, instead of proving an existing theory. The emergent theory acts as a working hypothesis and can be falsified through a deductive approach in other studies. Section 9.1.1.4 presents falsifiable hypotheses from our theory.

9 Conclusions

This chapter summarizes the contributions of this study to the theory and practices of Agile Software Development. The next section discusses the limitations of this study and of the emergent theory. In order to cope with the defined limitations, the last section then presents several suggestions for future work.

9.1 Research Contributions

The contributions of this study can be summarized into theoretical and practical contributions. In terms of theoretical contributions, this study contributes a theory of expertise coordination for Agile Software Development projects. This study contributes to practical Agile Software Development by the processes and strategies used to coordinate expertise. The next subsections describe the theoretical and practical contributions in detail.

9.1.1 Theoretical Contributions

This study contributes to software engineering theory regarding expertise coordination in Agile Software Development. This study introduces a theory about the expertise coordination process, coordinating outside expertise, and management support for coordinating expertise. A number of hypotheses are formulated from the emergent theory of expertise coordination and described at the end of this subsection.

9.1.1.1 Expertise Coordination Process

The thesis presents the theory of expertise coordination process, which describes how expertise is coordinated in Agile Software Development projects: *locating and recognizing expertise* and *accessing expertise.*

Locating and recognizing expertise enables Agile teams to identify the owner of expertise, quantify the level of expertise, and recognize that expertise. There is a need to facilitate identification of knowledge owners or experts in Agile teams, which is not explicitly reported in the literature [194]. Therefore, these findings address how Agile team members rely on each other in locating expertise: *communicating frequently, working closely together, declaring self-identified expertise, earning certification,* and *using an expertise directory* (see section 5.1). These findings have been presented and published in the XP2014 conference held in Rome, Italy [173].

The emergent approaches of locating and recognizing expertise are aligned with Agile values, principles, and practices. Our findings revealed that Agile values, principles, and practices provide spaces and opportunities for Agile teams to implement the emergent approaches of locating and recognizing expertise. Although, the last approach of locating and recognizing expertise, *using an expertise directory,* is also aligned with the first Agile value, *"Individual and interactions over processes and tools"* [59]. We found most Agile practitioners prefer communication instead of using an expertise directory, even though a directory would facilitate the process of

identifying and acknowledging expertise.

Locating and recognizing expertise leads to *accessing expertise*, where our findings revealed several approaches for accessing expertise: *embracing expert-novice relationships, engaging hands-on learning, running effective meetings, establishing discussion channels,* and *archiving explicit knowledge* (see section 5.2). These approaches are considered classifications of accessing expertise, where each approach comprises several techniques to retrieve available expertise and pull new expertise into Agile teams. These findings have been presented and published in the Agile conference held in Orlando, US [172].

Our findings indicated the existence of *apprenticeships* through *embracing expert-novice relationships* in Agile teams and how they differ from mentoring and coaching. Based on the differences, we identified several limitations of apprenticeships, which cause the minimal implementation of apprenticeships in Agile teams (see section 5.2.1.1). These findings contribute to the literature on Agile Software Development, which requires a further investigation into the proper implementation of apprenticeships in Agile Software Development [93].

The findings of the expertise coordination process contribute to the literature through two published international conferences as follows:

9.1.1.2 Coordinating Outside Expertise

This study gives new insights into the role of expertise outside Agile teams. Our theory suggests a new way of theorizing important aspects of coordinating external expertise with Agile teams. Our findings emphasize the relationships between external actors and Agile teams in coordinating external expertise through several strategies: *planning ahead, understanding Agile Mindset, retaining external expertise, ensuring consistency, treating outsourcers the same as in-house staff,* and *keeping everyone on the same page* (refer to Chapter 6). These findings fill a gap in the current research on Agile Software Development: Sharp and Robinson [194] have suggested further

investigation is needed to explore the interaction between Agile teams and roles outside the teams. We have presented and published a paper entitled "*Coordinating Expertise Outside Agile Teams*", to describe how Agile teams can deal with outside expertise at the XP2015 conference held in Helsinki, Finland, [172].

Coordinating outside expertise is based on the complexities or issues involved in coordinating expertise outside Agile teams: *availability, Agile mindset, knowledge retention, unknown working environment, stability,* and *effective communication.* These issues contribute to the literature by providing a new insight into factors that need to be considered when dealing with external specialists and outsourcers.

9.1.1.3 Management Support

This study contributes to the body of knowledge through the role of management in supporting expertise coordination in Agile Software Development projects. Management is responsible for supporting expertise coordination through emergent independent strategies: *self-selecting team, reforming performance appraisal,* and *embracing expertise sharing culture* (refer to Chapter 7).

Self-selecting teams are essential to supporting expertise coordination by increasing inter-dependencies between Agile team members, and ensuring a diverse range of knowledge and skills in teams. We revealed how to form self-selecting teams in supporting expertise coordination: *setting up cross-functional teams, selecting the right team,* and *balancing teams.* Through self-selecting teams, we found that several characteristics of teams, such as smaller team formation, as well as stable teams, lead to successful expertise coordination (see section 7.1). This study contributes to the body of knowledge by identifying the way self-selecting teams support expertise coordination.

Performance appraisal is vital to support expertise coordination by identifying gaps in team members' skills and competencies, and enabling

expertise development. Moving from traditional performance appraisal to an Agile performance appraisal is needed to adapt and align to Agile values, principles, and practices. Through this study, we revealed two major changes for reforming performance appraisal: *integrating individual and team performance assessment criteria,* and *shifting from quantitative to qualitative performance appraisal* (see section 7.2). These changes contribute to the literature in boosting the implementation of Agile-compliant performance appraisals. As Agile-compliant performance appraisal tends to influence a reward system, our findings contribute to the literature, as we suggest a proper balance between individual and team rewards for successful expertise sharing.

9.1.1.4 Working Hypotheses

The theory of expertise coordination is represented by using a theoretical code that consists of relationships between emergent categories, as depicted in Figure 9.1, highlighting the formulated hypotheses. Based on the structure of relationships in that theory, we have formulated 23 hypotheses from the emergent theory of expertise coordination, as follows:

- H1 : *locating and recognizing expertise will have a positive influence on accessing expertise.*
- H2 : *self-selecting teams will have a positive influence on locating and recognizing expertise*.
- H3 : *self-selecting teams will have a positive influence on accessing expertise.*
- H4 : *reforming performance appraisal will have a positive influence on locating and recognizing expertise.*
- H5 : *reforming performance appraisal will have a positive influence on accessing expertise.*

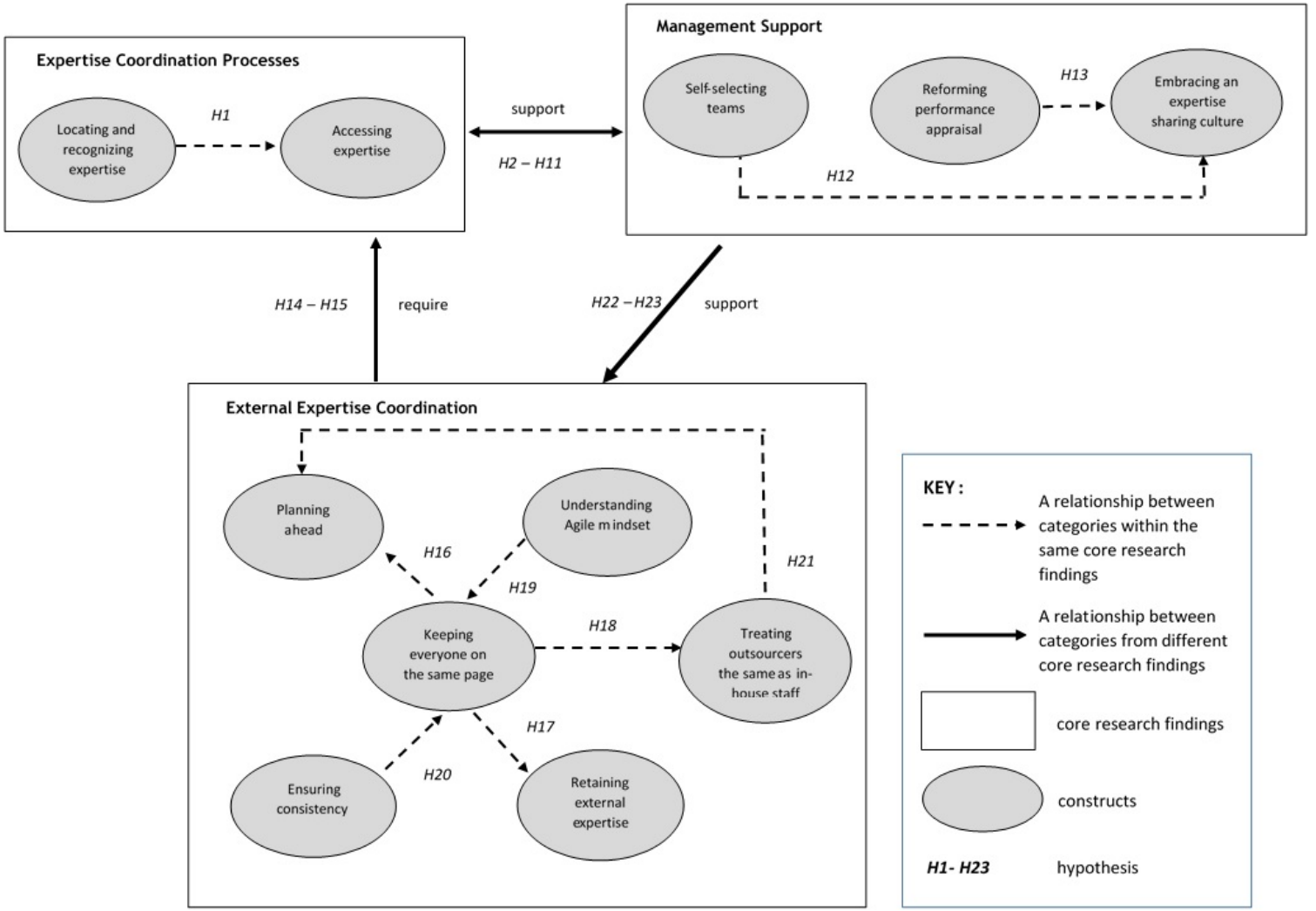


Figure 9.1: The Emergent Theoretical Code with Formulated Hypotheses.

- H6 : *embracing an expertise sharing culture will have a positive influence on locating and recognizing expertise.*
- H7 : *embracing an expertise sharing culture will have a positive influence on accessing expertise.*
- H8 : *locating and recognizing expertise will have a positive influence on embracing an expertise sharing culture.*
- H9 : *accessing expertise will have a positive influence on embracing an expertise sharing culture.*
- H10 : *locating and recognizing expertise will have a positive influence on reforming performance appraisal.*
- H11 : *locating and recognizing expertise will have a positive influence on self-selecting teams.*
- H12 : *self-selecting teams will have a positive influence on embracing an expertise sharing culture.*
- H13 : *reforming performance appraisal will have a positive influence on embracing an expertise sharing culture.*
- H14 : *locating and recognizing expertise will have a positive influence on treating outsourcers the same as in-house staff.*
- H15 : *accessing expertise will have a positive influence on retaining external expertise.*
- H16 : *keeping everyone on the same page will have a positive influence on planning ahead.*
- H17 : *keeping everyone on the same page will have a positive influence on retaining external expertise.*

- H18 : *keeping everyone on the same page will have a positive influence on treating outsourcers the same as in-house staff*.
- H19 : *understanding Agile mindset will have a positive influence on keeping everyone on the same page*.
- H20 : *ensuring consistency will have a positive influence on keeping everyone on the same page*.
- H21 : *treating outsourcers the same as in-house staff will have a positive influence on planning ahead*.
- H22 : *self-selecting teams will have a positive influence on ensuring consistency*.
- H23 : *embracing an expertise sharing culture will have a positive influence on keeping everyone on the same page.*

These working hypotheses can be tested through a deductive approach in future studies.

9.1.2 Practical Contributions

This study makes practical contributions to Agile Software Development. We enrich the existing approaches in identifying the owner of expertise, and recognizing that particular expertise. The approaches of *locating and recognizing expertise* can be used by Agile practitioners when they are seeking expertise in accomplishing their tasks. The choice of approaches vary depending on practitioners' preferences, and when the approach is suited to be applied in teams (see section 5.1).

In terms of *accessing expertise*, Agile practitioners can adopt the approaches for accessing expertise. We revealed commonly used approaches, as well as new insights about retrieving available expertise and pulling new expertise into Agile teams, such as apprenticeships, role meetings, and document management tools (see section 5.2).

Agile practitioners can obtain benefits from external expertise coordination strategies (refer to Chapter 6). These strategies can be used directly by Agile practitioners when they are dealing with external actors such as external specialists and outsourcers. Adopting these strategies tends to mitigate several issues which arise when coordinating expertise outside Agile teams. This thesis provides Agile practitioners with in-depth understanding of some complexities or issues involved in coordinating expertise of external specialists and outsourcers.

This study also provides external specialists and outsourcers with important insights into how to coordinate their expertise with Agile teams. It is impossible to utilize external expertise without effective cooperation and support from external specialists and outsourcers. Therefore, external specialists and outsourcers need to ensure their availability when needed, understand the Agile Mindset, retain their expertise in Agile teams, easily adapt to a new working environment, work in accordance with a particular standard work, have been treated the same as in-house staff, and keep themselves on the same page.

This study also has implications for management in supporting expertise coordination. Management needs to emphasize self-selecting teams, rather than just selecting the available people to join in a particular software project (see section 7.3). It is vital for management to facilitate the process of self-selecting teams in order to optimize successful expertise coordination.

On the other hand, Agile practitioners can consider several factors in choosing the right team: *types of work, team members,* and *knowledge sharing* (see section 7.1.2). Agile practitioners are advised to prioritize knowledge sharing in supporting expertise coordination.

This study suggests management reform performance appraisal, in order to ensure it is aligned with Agile values, principles, and practices (see section 7.2). Management should integrate individual and team performance assessment criteria, and emphasize qualitative measurement through feedback. These changes lead to the improvement of performance

appraisal that tend to support expertise coordination.

This study also proposes management set up a reward system that is suitably designed for Agile teams. A proper balance between individual and team rewards is essential for successful expertise sharing, which influences successful expertise coordination.

9.2 Research Limitations

Although this study generates important findings of expertise coordination in the context of Agile Software Development, we acknowledge several research limitations.

The first research limitation is related to generalization. The aim of this study is not to generalize our findings to different contexts, but to discover a theory of the phenomena studied. Therefore, the emergent theory does not need further testing, but it can be falsified through a deductive approach in other studies. The theory of expertise coordination is only applicable within the Agile Software Development context, and further study is needed to ensure the applicability of our findings within other contexts.

The next research limitation is the applicability of the theory to all forms of Agile project. We recruited participants without restriction as to which Agile methods have been used in Agile projects. The theory, however, is mostly based on research evidence from Agile projects using Scrum and XP practices. Since Scrum and XP are currently the dominant Agile methods, we believe the theory represents expertise coordination from the various points of view of those Agile methods.

The applicability of the theory is not specific to co-located or distributed Agile projects. We have recruited participants without considering co-located or distributed Agile teams. Therefore, this study represents the theory of expertise coordination in a broad spectrum of Agile teams.

A further research limitation is concerned with the range of research participants. The participants were selected and recruited based on ac-

cessibility, thus mostly based in Wellington, New Zealand, and several locations of Agile conferences and workshops. This restricted the range of participants, particularly their working practices, culture, and experiences, which must influence our findings.

The limitation of the range of participants was also caused by a small number of interviews conducted with external specialists and outsourcers. It was quite hard to recruit external specialists and outsourcers due to their limited involvement in Agile teams. Our findings, however, have been strengthened with supporting interviews with Agile team members, since we need to discover how Agile teams interact with external specialists and outsourcers in coordinating expertise outside Agile teams.

There is no customer involvement in our study, which also contributes to a limitation of the range of participants. Agile Software Development requires Agile teams to interact extensively with customers, and we believe this tends to influence expertise coordination. We attempted to recruit customers in our study, however, none were willing to participate. Since this study has recruited other Agile roles, including external specialists and outsourcers, this might not be a major limitation.

We have identified the need for interconnections between team members in Agile projects. Therefore, we had to interview more than one participant in the same Agile project in order to affirm the consistency of findings. Unfortunately, we only managed to recruit a few participants from the same project. This is also limited to the range of participants in our study.

In order to mitigate the limitations of the range of research participants, constant comparison and theoretical sampling were used to determine future data required, based on the current data analysis of this study. This is important to ensure the comprehensive nature of data.

A further research limitation is related to data triangulation. We managed to carry out observations at only one company, due to difficulties and challenges in getting Agile Software Development companies to take part in this study. We conducted six observational sessions on two different

software development projects over two months, and consequently we had limited observation data to strengthen the triangulation.

Another research limitation is related to research analysis. Member checks were solely by the researchers of this study. That is to say we relied on participants' input for confirming data and interpretation, without appointing independent people to clarify and confirm our research data and interpretation.

The last research limitation is the researcher's ability in conducting a Grounded Theory study. A lack of experience in Grounded Theory meant we spent more time in developing an understanding of Grounded Theory, and analyzing the collected data until the core category emerged.

Even though several research limitations have been identified, not all of them could be mitigated during the research process. However, we have tried to mitigate the limitations as much as we could. We hope further studies with other research methods could address these limitations.

9.3 Future Work

Future work is essential to confirm the theory of this study by using other research approaches, form a formal theory, and enhance the efficacy of the theory. In particular, future research should aim to verify the theory of expertise coordination by using other research approaches, either qualitative or quantitative. Through case study or action research, future researchers could generate more hypotheses of expertise coordination, which were not discovered throughout this study. These qualitative approaches and other quantitative methods also can be used to test the formulated hypotheses derived from this study (subsubsection 9.1.1.4). For instance, a survey method can be employed to verify a significant relationship between *self-selecting teams* and *locating and recognizing expertise*.

This study has generated a substantive theory, which specifically focuses on a particular phenomenon: expertise coordination of Agile Software

Development. This substantive theory can be extended to form a formal theory. The substantive theory can be replicated within different contexts of study such as non-Agile Software Development, distributed Agile Software Development, and healthcare organizations. Constant comparison could be used to compare the different substantive theories and lead to the development of a formal theory, which would explicate abstract and general expertise coordination.

There is potential for further Grounded Theory research to explore expertise coordination in Agile Software Development Projects in more depth. Further research could explore more strategies and processes involved in coordinating expertise in Agile Software Development projects. For instance, the concepts of expertise coordination process need better conceptualization since the concepts might not adequately cover the process involved.

Our study also leads to further investigation on aspects related to expertise coordination as follows:

- A further investigation on the impact of the knowledge sharing factors in self-selecting teams.
- A further investigation on the proper implementation of apprenticeships in Agile Software Development.

Bibliography

[1] ABRAHAMSSON, P., CONBOY, K., AND WANG, X. 'lots done, more to do': the current state of agile systems development research. *Palgrave Macmillan Ltd.,* (2009).

[2] ABRAHAMSSON, P., SALO, O., RONKAINEN, J., AND WARSTA, J. Agile software development methods: Review and analysis, 2002.

[3] ADOLPH, S., HALL, W., AND KRUCHTEN, P. Using grounded theory to study the experience of software development. *Empirical Software Engineering 16*, 4 (2011), 487–513.

[4] AHMAD, R., AND BUJANG, S. Issues and challenges in the practice of performance appraisal activities in the 21st century. *International Journal of Education and Research 1*, 4 (2013), 7.

[5] AKGÜN, A. E., BYRNE, J., KESKIN, H., LYNN, G. S., AND IMAMOGLU, S. Z. Knowledge networks in new product development projects: A transactive memory perspective. *Information & management 42*, 8 (2005), 1105–1120.

[6] AKGUN, A. E., BYRNE, J. C., KESKIN, H., AND LYNN, G. S. Transactive memory system in new product development teams. *Engineering Management, IEEE Transactions on 53*, 1 (2006), 95–111.

[7] ALLAN, G. A critique of using grounded theory as a research method. *Electronic Journal of Business Research Methods 2*, 1 (2003), 1–10.

[8] ALNAJI, L., AND SALAMEH, H. Performance-measurement framework to evaluate software engineers for agile software-development

methodology. *European Journal of Business and Management 7*, 2 (2015), 183–190.

[9] AMBLER, S. W. Generalizing specialists: Improving your IT career skills. `http://www.agilemodeling.com/essays/generalizingSpecialists.htm`, 2003. [Online; accessed 11-March-2013].

[10] ARENAS, A. E., AND BARRERA-SANABRIA, G. Applying the mas-commonkads methodology to the flights reservation problem: Integrating coordination and expertise. In *Proceedings of the Fifth Joint Conference on Knowledge-Based Software Engineering (JCKBSE 2002* (2002), Citeseer.

[11] ARGOTE, L. Input uncertainty and organizational coordination in hospital emergency units. *Administrative science quarterly* (1982), 420–434.

[12] ARGOTE, L. Organizational learning research: Past, present and future. *Management Learning 42*, 4 (2011), 439–446.

[13] AWAD, M. A comparison between agile and traditional software development methodologies. *University of Western Australia* (2005).

[14] BAKER, S. E., AND EDWARDS, R. *How many qualitative interviews is enough? Expert voices and early career reflections on sampling and cases in qualitative research*. National Centre for Research Methods, 2012.

[15] BANNERMAN, P. L., HOSSAIN, E., AND JEFFERY, R. Scrum practice mitigation of global software development coordination challenges: A distinctive advantage? In *System Science (HICSS), 2012 45th Hawaii International Conference on* (2012), IEEE, pp. 5309–5318.

[16] BARTHELMESS, P. Collaboration and coordination in process-centered software development environments: a review of the literature. *Information and Software Technology 45*, 13 (2003), 911–928.

[17] BASS, M. Monitoring GSD projects via shared mental models: a suggested approach. In *Proceedings of the 2006 international workshop on Global software development for the practitioner* (2006), ACM, pp. 34–37.

[18] BATRA, D., XIA, W., VANDERMEER, D., AND DUTTA, K. Balancing agile and structured development approaches to successfully manage large distributed software projects: A case study from the cruise line industry. *Communications of the Association for Information Systems 27*, 1 (2010), 21.

[19] BECK, K. *Extreme programming explained: embrace change*. Addison-Wesley Professional, 1999.

[20] BECK, K., AND ANDRES, C. *Extreme programming explained: embrace change, 2nd Edition*. Addison-Wesley Professional, 2004.

[21] BERTONI, M., AND CHIRUMALLA, K. Leveraging web 2.0 in new product development: Lessons learned from a cross-company study. *J. UCS 17*, 4 (2011), 548–564.

[22] BIELACZYC, K., AND COLLINS, A. Learning communities in classrooms: A reconceptualization of educational practice. *Instructional-design theories and models: A new paradigm of instructional theory 2* (1999), 269–292.

[23] BIRKS, M., AND MILLS, J. *Grounded theory: a practical guide*. Sage Publications Limited, 2011.

[24] BOEHM, B., AND MOBASSER, S. K. System thinking: educating t-shaped software engineers. In *2015 IEEE/ACM 37th IEEE International Conference on Software Engineering* (2015), vol. 2, IEEE, pp. 333–342.

[25] BOWEN, G. A. Document analysis as a qualitative research method. *Qualitative research journal 9*, 2 (2009), 27–40.

[26] BROOKS, N. Understanding IT outsourcing and its potential effects on it workers and their environment. *The Journal of Computer Information Systems 46*, 4 (2006), 46.

[27] CAETANO, A., POMBINHO, J., AND TRIBOLET, J. Representing organizational competencies. In *Proceedings of the 2007 ACM symposium on Applied computing* (2007), ACM, pp. 1257–1262.

[28] CALDWELL, B. S. Knowledge sharing and expertise coordination of event response in organizations. *Applied ergonomics 39*, 4 (2008), 427–438.

[29] CAO, L., AND RAMESH, B. Agile software development: Ad hoc practices or sound principles? *IT professional 9*, 2 (2007), 41–47.

[30] CHAO, J., AND ATLI, G. Critical personality traits in successful pair programming. In *Agile Conference, 2006* (2006), IEEE, p. 5.

[31] CHARMAZ, K. *Constructing grounded theory: A practical guide through qualitative analysis*. Sage Publications Limited, 2006.

[32] CHAU, T., AND MAURER, F. Knowledge sharing in agile software teams. In *Logic versus approximation*. Springer, 2004, pp. 173–183.

[33] CHAU, T., MAURER, F., AND MELNIK, G. Knowledge sharing: Agile methods vs. tayloristic methods. In *Enabling Technologies: Infrastructure for Collaborative Enterprises, 2003. WET ICE 2003. Proceedings. Twelfth IEEE International Workshops on* (2003), IEEE, pp. 302–307.

[34] CHO, J. Issues and challenges of agile software development with scrum. *Issues in Information Systems 9*, 2 (2008), 188–195.

[35] CHOW, T., AND CAO, D. A survey study of critical success factors in agile software projects. *Journal of Systems and Software 81*, 6 (2008), 961–971.

[36] CLOKIE, K. Test Strategy Retrospectives. `http://katrinatester.blogspot.my/2014/04/test-strategy-retrospective.html`, 2014. [Online; accessed 1-January-2015].

[37] COCKBURN, A. Agile software development joins the 'would–be' crowd. *Cutter IT Journal 15*, 1 (2002), 6–12.

[38] COCKBURN, A. *Crystal clear: a human-powered methodology for small teams*. Addison-Wesley Professional, 2005.

[39] COCKBURN, A., AND HIGHSMITH, J. Agile software development, the people factor. *Computer 34*, 11 (2001), 131–133.

[40] COHN, M. *Succeeding with agile: software development using Scrum*. Pearson Education, 2010.

[41] COLEMAN, G., AND O'CONNOR, R. Using grounded theory to understand software process improvement: A study of irish software product companies. *Information and Software Technology 49*, 6 (2007), 654–667.

[42] CORBIN, J. M., AND STRAUSS, A. Grounded theory research: Procedures, canons, and evaluative criteria. *Qualitative sociology 13*, 1 (1990), 3–21.

[43] COYLE, S., AND CONBOY, K. People over process: key people challenges in agile development. *IEEE* (2010).

[44] CRESWELL, J. W. *Qualitative inquiry and research design: Choosing among five approaches*. Sage, 2012.

[45] CROWSTON, K., ANNABI, H., HOWISON, J., AND MASANGO, C. Effective work practices for software engineering: free/libre open source software development. In *Proceedings of the 2004 ACM workshop on Interdisciplinary software engineering research* (2004), ACM, pp. 18–26.

[46] DE O MELO, C., S CRUZES, D., KON, F., AND CONRADI, R. Interpretative case studies on agile team productivity and management. *Information and Software Technology 55*, 2 (2013), 412–427.

[47] DENNING, S. Why agile can be a game changer for managing continuous innovation in many industries. *Strategy & Leadership 41*, 2 (2013), 5–11.

[48] DENZIN, N. K., AND LINCOLN, Y. S. *The SAGE handbook of qualitative research*. Sage Publications, Incorporated, 2011.

[49] DESSAI, K., AND KAMAT, M. Application of social media for tracking knowledge in agile software projects. *Available at SSRN 2018845* (2012).

[50] DORAIRAJ, S. *The Theory of One Team: Agile Software Development with Distributed Teams*. Victoria University of Wellington, 2013.

[51] DORAIRAJ, S., NOBLE, J., AND ALLAN, G. Agile software development with distributed teams: Senior management support. In *Global Software Engineering (ICGSE), 2013 IEEE 8th International Conference on* (2013), IEEE, pp. 197–205.

[52] EHRLICH, K., AND CHANG, K. Leveraging expertise in global software teams: Going outside boundaries. In *Global Software Engineering, 2006. ICGSE'06. International Conference on* (2006), IEEE, pp. 149–158.

[53] ELSHAMY, A., AND ELSSAMADISY, A. Divide after you conquer: an agile software development practice for large projects. In *Extreme Programming and Agile Processes in Software Engineering*. Springer, 2006, pp. 164–168.

[54] FARAJ, S., AND SPROULL, L. Coordinating expertise in software development teams. *Management Science* (2000), 1554–1568.

[55] FARAJ, S., AND XIAO, Y. Coordination in fast-response organizations. *Management science 52*, 8 (2006), 1155–1169.

[56] FEDEROFF, M., AND COURAGE, C. Successful user experience in an agile enterprise environment. In *Human interface and the management of information. Designing information environments*. Springer, 2009, pp. 233–242.

[57] FENGJIE, A., FEI, Q., AND XIN, C. Knowledge sharing and web-based knowledge-sharing platform. In *E-Commerce Technology for Dynamic E-Business, 2004. IEEE International Conference on* (2004), IEEE, pp. 278–281.

[58] FERNIE, S., GREEN, S. D., WELLER, S. J., AND NEWCOMBE, R. Knowledge sharing: context, confusion and controversy. *International Journal of Project Management 21*, 3 (2003), 177–187.

[59] FOWLER, M., AND HIGHSMITH, J. The agile manifesto. *Software Development 9*, 8 (2001), 28–35.

[60] GARCÍA, J., AMESCUA, A., SÁNCHEZ, M.-I., AND BERMÓN, L. Design guidelines for software processes knowledge repository development. *Information and Software Technology 53*, 8 (2011), 834–850.

[61] GARRETT, S., CALDWELL, B., HARRIS, E., AND GONZALEZ, M. Six dimensions of expertise: a more comprehensive definition of cognitive expertise for team coordination. *Theoretical Issues in Ergonomics Science 10*, 2 (2009), 93–105.

[62] GARRETT, S. K., CALDWELL, B. S., AND COLLINS, S. T. Supporting expertise coordination in multidisciplinary project teams. In *Proceedings of the Human Factors and Ergonomics Society Annual Meeting* (2009), vol. 53, SAGE Publications, pp. 1008–1012.

[63] GERARD, J. G. Measuring knowledge source tacitness and explicitness: A comparison of paired items. In *Proceedings: 5th Annual Organizational Learning and Knowledge Conference* (2003).

[64] GIBBS, J. Dialectics in a global software team: Negotiating tensions across time, space, and culture. *Human Relations 62*, 6 (2009), 905–935.

[65] GITTELL, J. H., SEIDNER, R., AND WIMBUSH, J. A relational model of how high-performance work systems work. *Organization science 21*, 2 (2010), 490–506.

[66] GLASER, B. G. *Theoretical sensitivity: Advances in the methodology of grounded theory*, vol. 2. Sociology Press Mill Valley, CA, 1978.

[67] GLASER, B. G. *Emergence vs forcing: Basics of grounded theory analysis*. Sociology Press, 1992.

[68] GLASER, B. G. *Doing grounded theory: Issues and discussions*. Sociology Press, 1998.

[69] GLASER, B. G., AND HOLTON, J. Remodeling grounded theory. In *Forum Qualitative Sozialforschung/Forum: Qualitative Social Research* (2004), vol. 5.

[70] GLASER, B. G., AND HOLTON, J. Basic social processes. *Grounded Theory Review 4*, 3 (2005), 1–29.

[71] GLASER, B. G., AND STRAUSS, A. L. *The discovery of grounded theory: Strategies for qualitative research*. Aldine de Gruyter, 1967.

[72] GRISHAM, P. S., AND PERRY, D. E. Customer relationships and extreme programming. In *ACM SIGSOFT Software Engineering Notes* (2005), vol. 30, ACM, pp. 1–6.

[73] GUBA, E. G., AND LINCOLN, Y. S. *Fourth generation evaluation*. Sage, 1989.

[74] GUBA, E. G., AND LINCOLN, Y. S. Competing paradigms in qualitative research. *Handbook of qualitative research 2* (1994), 163–194.

[75] HANCOCK, B. *An introduction to qualitative research*. Trent focus group, 1998.

[76] HANLY, S., WAI, L., MEADOWS, L., AND LEATON, R. Agile coaching in British Telecom: Making strawberry jam. In *Agile Conference, 2006* (2006), IEEE, pp. 9–pp.

[77] HASSAN, A. M., AND ELSSAMADISY, A. Extreme programming and database administration: Problems, solutions, and issues, 2002.

[78] HERBSLEB, J. D. Global software engineering: The future of socio-technical coordination. In *2007 Future of Software Engineering* (2007), IEEE Computer Society, pp. 188–198.

[79] HERLING, R. Operational definitions of expertise and competence. *Advances in developing human resources 2*, 1 (2000), 8–21.

[80] HERNANDEZ, C. A. Theoretical coding in grounded theory methodology. *Grounded Theory Review 8*, 3 (2009).

[81] HIGHSMITH, J., AND HIGHSMITH, J. A. *Agile project management: creating innovative products*. Addison-Wesley Professional, 2009.

[82] HIGHSMITH, J. A. *Adaptive software development*. Dorset House New York, 2000.

[83] HODA, R. Self-organizing agile teams: A grounded theory. *Phd Thesis, Victoria University of Wellington* (2011).

[84] HODA, R., NOBLE, J., AND MARSHALL, S. Organizing self-organizing teams. In *Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering-Volume 1* (2010), ACM, pp. 285–294.

[85] HODA, R., NOBLE, J., AND MARSHALL, S. Developing a grounded theory to explain the practices of self-organizing agile teams. *Empirical Software Engineering 17*, 6 (2012), 609–639.

[86] HOLE, S., AND MOE, N. B. A case study of coordination in distributed agile software development. In *Software process improvement*. Springer, 2008, pp. 189–200.

[87] HOLLINGSHED, A. B. Perceptions of expertise and transactive memory in work relationship. *Group Processes Intergroup Relations 3*, 3 (2000), 257–267.

[88] HOLZ, H., AND MAURER, F. Knowledge management support for distributed agile software processes. *Advances in Learning Software Organizations* (2003), 60–80.

[89] HOSSAIN, E., BABAR, M. A., AND PAIK, H.-Y. Using scrum in global software development: a systematic literature review. In *Global Software Engineering, 2009. ICGSE 2009. Fourth IEEE International Conference on* (2009), Ieee, pp. 175–184.

[90] ISHENKO, O. Outsourcing of software development. *Humbold University Berlin, Wirtschaftwissenschaftliche Fakultaet, study* (2005).

[91] JACKLING, B., AND SULLIVAN, C. Financial planners in australia: an evaluation of gaps in technical and behavioral skills. *Financial Services Review 16*, 3 (2007), 211.

[92] JANEV, V., DUDUKOVIC, J., AND VRANEŠ, S. Semantic web based integration of knowledge resources for expertise finding. *International Journal of Enterprise Information Systems (IJEIS) 5*, 4 (2009), 53–70.

[93] JUDY, K. H. Agile principles and ethical conduct. In *System Sciences, 2009. HICSS'09. 42nd Hawaii International Conference on* (2009), IEEE, pp. 1–8.

[94] JULI, T. *The Power and Illusion of Self-organizing Teams*. Project Management Institute, 2012.

[95] KEITH, M., GOUL, M., DEMIRKAN, H., NICHOLS, J., AND MITCHELL, M. C. Contextualizing knowledge management readiness to support change management strategies. In *Proceedings of the 39th Annual Hawaii International Conference on System Sciences (HICSS'06)* (2006), vol. 7, IEEE, pp. 152a–152a.

[96] KIRK, J., AND MILLER, M. L. *Reliability and validity in qualitative research*. Sage, 1986.

[97] KOLAWA, A. Software certification debate: Certification will do more harm than good. *Computer*, 6 (2002), 34–35.

[98] KOLB, S. M. Grounded theory and the constant comparative method: valid research strategies for educators. *Journal of Emerging Trends in Educational Research and Policy Studies 3*, 1 (2012), 83–86.

[99] KOLLMANN, J., SHARP, H., AND BLANDFORD, A. The importance of identity and vision to user experience designers on agile projects. In *Agile Conference, 2009. AGILE'09.* (2009), IEEE, pp. 11–18.

[100] KOTLARSKY, J., SCARBROUGH, H., AND OSHRI, I. Coordinating expertise across knowledge boundaries in offshore-outsourcing projects: The role of codification. *Management information systems quarterly 38*, 2 (2014), 607–627.

[101] KOTLARSKY, J., VAN FENEMA, P. C., AND WILLCOCKS, L. P. Developing a knowledge-based perspective on coordination: The case of global software projects. *Information & Management 45*, 2 (2008), 96–108.

[102] KRAUT, R., AND STREETER, L. Coordination in software development. *Communications of the ACM 38*, 3 (1995), 69–81.

[103] KUA, P. The retrospective handbook. *E–book available at: https://leanpub. com/the–retrospective–handbook* (2013).

[104] KUCHINKE, K. The status of the theory and the literature. *Performance Improvement Quarterly 10* (1997), 72–86.

[105] KUTAY, C., AND AURUM, A. Knowledge transformation for education in software engineering. *International Journal of Mobile Learning and Organisation 1*, 1 (2006), 58–80.

[106] LALSING, V., KISHNAH, S., AND PUDARUTH, S. People factors in agile software development and project management. *International Journal of Software Engineering & Applications (IJSEA) 3*, 1 (2012), 117–137.

[107] LAPHAM, M. A. Dod agile adoption: Necessary considerations, concerns, and changes. Tech. rep., DTIC Document, 2012.

[108] LARMAN, C., AND BASILI, V. R. Iterative and incremental developments. a brief history. *Computer 36*, 6 (2003), 47–56.

[109] LAW, A., AND CHARRON, R. Effects of agile practices on social factors. In *ACM SIGSOFT Software Engineering Notes* (2005), vol. 30, ACM, pp. 1–5.

[110] LAYMAN, L., WILLIAMS, L., DAMIAN, D., AND BURES, H. Essential communication practices for extreme programming in a global software development team. *Information and software technology 48*, 9 (2006), 781–794.

[111] LÁZARO, M., AND MARCOS, E. An approach to the integration of qualitative and quantitative research methods in software engineering research. In *2nd International Workshop on Philosophical Foundations of Information Systems Engineering (PHISE'06) LNCS. Springer-Verlag, Berlin* (2006).

[112] LAZENBATT, A., ELLIOTT, N., ET AL. *How to recognise a'quality'grounded theory research study*. Australian Nursing Federation, 2005.

[113] LEE, G., AND XIA, W. Toward agile: An integrated analysis of quantitative and qualitative field data on software development agility. *Mis Quarterly 34*, 1 (2010).

[114] LEECH, N. L., AND ONWUEGBUZIE, A. J. Beyond constant comparison qualitative data analysis: Using Nvivo. *School Psychology Quarterly 26*, 1 (2011), 70.

[115] LEFFINGWELL, D. *Agile software requirements: lean requirements practices for teams, programs, and the enterprise*. Addison-Wesley, 2010.

[116] LEONARDI, P. M., AND TREEM, J. W. Knowledge management technology as a stage for strategic self-presentation: Implications for knowledge sharing in organizations. *Information and Organization 22*, 1 (2012), 37–59.

[117] LEVESQUE, L. L., WILSON, J. M., AND WHOLEY, D. R. Cognitive divergence and shared mental models in software development project teams. *Journal of Organizational Behavior 22*, 2 (2001), 135–144.

[118] LEWIS, K. Measuring transactive memory systems in the field: scale development and validation. *Journal of Applied Psychology 88*, 4 (2003), 587.

[119] LI, Y., AND MAEDCHE, A. *Formulating effective coordination strategies in agile global software development teams*. 2012.

[120] LIN, J., MIAO, C., SHEN, Z., AND SUN, W. Goal oriented agile unified process (goaup): An educational case study. In *2013 International Conference on Software Engineering and Computer Science* (2013), Atlantis Press.

[121] LINCOLN, Y. S., AND GUBA, E. G. *Naturalistic inquiry*, vol. 75. Sage, 1985.

[122] LINDSTROM, L., AND JEFFRIES, R. Extreme programming and agile software development methodologies. *Information Systems Management 21*, 3 (2004), 41–52.

[123] LINDVALL, M., BASILI, V., BOEHM, B., COSTA, P., DANGLE, K., SHULL, F., TESORIERO, R., WILLIAMS, L., AND ZELKOWITZ, M. Empirical findings in agile methods. In *Extreme Programming and Agile Methods?XP/Agile Universe 2002*. Springer, 2002, pp. 197–207.

[124] LITTLEPAGE, G., ROBISON, W., AND REDDINGTON, K. Effects of task experience and group experience on group performance, member ability, and recognition of expertise. *Organizational Behavior and Human Decision Processes 69*, 2 (1997), 133–147.

[125] LOHAN, G., CONBOY, K., AND LANG, M. Beyond budgeting and agile software development: a conceptual framework for the performance management of agile software development teams. In *ICIS* (2010), p. 162.

[126] LOMBORG, K., AND KIRKEVOLD, M. Truth and validity in grounded theory–a reconsidered realist interpretation of the criteria: fit, work, relevance and modifiability. *Nursing Philosophy 4*, 3 (2003), 189–200.

[127] MACKENZIE, A., AND MONK, S. From cards to code: How extreme programming re-embodies programming as a collective practice. *Computer Supported Cooperative Work (CSCW) 13*, 1 (2004), 91–117.

[128] MAJCHRZAK, A., JARVENPAA, S., AND HOLLINGSHEAD, A. Coordinating expertise among emergent groups responding to disasters. *Organization Science 18*, 1 (2007), 147–161.

[129] MALONE, T., AND CROWSTON, K. The interdisciplinary study of coordination. *ACM Computing Surveys (CSUR) 26*, 1 (1994), 87–119.

[130] MALONE, T. W., ET AL. *What is coordination theory?* Citeseer, 1988.

[131] MARTIN, A., BIDDLE, R., AND NOBLE, J. The XP customer role in practice: Three studies. In *Agile Development Conference, 2004* (2004), IEEE, pp. 42–54.

[132] MARTINI, A., PARETO, L., AND BOSCH, J. Teams interactions hindering short-term and long-term business goals. In *Continuous Software Engineering*. Springer, 2014, pp. 51–65.

[133] MARUPING, L., ZHANG, X., AND VENKATESH, V. Role of collective ownership and coding standards in coordinating expertise in software project teams. *European Journal of Information Systems 18*, 4 (2009), 355–371.

[134] MCCONNELL, S. *Rapid development: taming wild software schedules*. Pearson Education, 1996.

[135] MCGEE-LENNON, M., SMEATON, A., AND BREWSTER, S. Designing home care reminder systems: lessons learned through co-design with older users. In *Pervasive Computing Technologies for Healthcare (PervasiveHealth), 2012 6th International Conference on* (2012), IEEE, pp. 49–56.

[136] MELNIK, G., AND MAURER, F. Direct verbal communication as a catalyst of agile knowledge sharing. In *Agile Development Conference, 2004* (2004), IEEE, pp. 21–31.

[137] MELO, C. D. O., CRUZES, D. S., KON, F., AND CONRADI, R. Interpretative case studies on agile team productivity and management. *Information and Software Technology 55*, 2 (2013), 412–427.

[138] MEZICK, D. J. Organizational learning with open space. In *Agile Conference (AGILE), 2013* (2013), IEEE, pp. 142–149.

[139] MIDDLETON, P., AND JOYCE, D. Lean software management: BBC worldwide case study. *Engineering Management, IEEE Transactions on 59*, 1 (2012), 20–32.

[140] MILES, M. B., AND HUBERMAN, A. M. Qualitative data analysis: An expanded sourcebook. thousands oaks, 1994.

[141] MILLER, G. A. The magical number seven, plus or minus two: some limits on our capacity for processing information. *Psychological review 63*, 2 (1956), 81.

[142] MINICHIELLO, V., ARONI, R., HAYS, T., ET AL. *In-depth interviewing: Principles, techniques, analysis*. Pearson Education Australia, 2008.

[143] MINTZBERG, H. Structure in 5's: A synthesis of the research on organization design. *Management science 26*, 3 (1980), 322–341.

[144] MISHRA, D., AND MISHRA, A. Effective communication, collaboration, and coordination in extreme programming: Human-centric perspective in a small organization. *Human Factors and Ergonomics in Manufacturing & Service Industries 19*, 5 (2009), 438–456.

[145] MISHRA, D., MISHRA, A., AND OSTROVSKA, S. Impact of physical ambiance on communication, collaboration and coordination in agile software development: An empirical evaluation. *Information and software Technology 54*, 10 (2012), 1067–1078.

[146] MISRA, S., KUMAR, V., AND KUMAR, U. Identifying some important success factors in adopting agile software development practices. *Journal of Systems and Software 82*, 11 (2009), 1869–1890.

[147] MOCKUS, A., AND HERBSLEB, J. Expertise browser: a quantitative approach to identifying expertise. In *Proceedings of the 24th International Conference on Software Engineering* (2002), ACM, pp. 503–512.

[148] MOE, N. B., DINGSØYR, T., AND DYBÅ, T. Understanding self-organizing teams in agile software development. In *19th Australian Conference on Software Engineering (aswec 2008)* (2008), IEEE, pp. 76–85.

[149] MOE, N. B., DINGSØYR, T., AND DYBÅ, T. A teamwork model for understanding an agile team: A case study of a scrum project. *Information and Software Technology 52*, 5 (2010), 480–491.

[150] MOE, N. B., ŠMITE, D., ŠĀBLIS, A., BÖRJESSON, A.-L., AND ANDRÉASSON, P. Networking in a large-scale distributed agile project. In *International Symposium on Empirical Software Engineering and Measurement* (2014), ACM.

[151] MUSHTAQ, Z., AND QURESHI, M. R. J. Novel hybrid model: Integrating scrum and xp. *International Journal of Information Technology and Computer Science (IJITCS) 4*, 6 (2012), 39.

[152] MYERS, M. D. Qualitative research in information systems. *Management Information Systems Quarterly 21* (1997), 241–242.

[153] NARAYANAN, S., BALASUBRAMANIAN, S., AND SWAMINATHAN, J. M. A matter of balance: Specialization, task variety, and individual learning in a software maintenance environment. *Management Science 55*, 11 (2009), 1861–1876.

[154] NARAYANAN, S., BALASUBRAMANIAN, S., AND SWAMINATHAN, J. M. Managing outsourced software projects: An analysis of project performance and customer satisfaction. *Production and Operations Management 20*, 4 (2011), 508–521.

[155] NERUR, S., MAHAPATRA, R., AND MANGALARAJ, G. Challenges of migrating to agile methodologies. *Communications of the ACM 48*, 5 (2005), 72–78.

[156] NOORI, S., HOSSEINI, S. H., AND BAKHSHA, A. Human performance factors in the evaluation of virtual organizations. *International Journal of Business and Management 4*, 2 (2009), p41.

[157] NUWANGI, S. M., SEDERA, D., AND MURPHY, G. *Multi-Level Knowledge Transfer In Software Development Outsourcing Projects: The Agency Theory View*. 2012.

[158] OLSSON, H. H., BOSCH, J., AND ALAHYARI, H. Towards R&D as innovation experiment systems: A framework for moving beyond agile software development. In *Proceedings of the IASTED* (2013), pp. 798–805.

[159] OMAR, M., SYED-ABDULLAH, S.-L., AND YASIN, A. The impact of agile approach on software engineering teams. *American Journal of Economics and Business Administration 3*, 1 (2011), 12.

[160] ONIONS, P. E. Grounded theory applications in reviewing knowledge management literature. In *Methodological issues and ethical considerations' conference* (2006), Citeseer.

[161] ORLIKOWSKI, W. J., AND BAROUDI, J. J. Studying information technology in organizations: Research approaches and assumptions. *Information systems research 2*, 1 (1991), 1–28.

[162] PAASIVAARA, M., VÄÄTTÄNEN, O., HALLIKAINEN, M., AND LASSENIUS, C. Supporting a large-scale lean and agile transformation by defining common values. In *International Conference on Agile Software Development* (2014), Springer, pp. 73–82.

[163] PALMER, S. R., AND FELSING, M. *A practical guide to feature-driven development*. Pearson Education, 2001.

[164] PALS, N., STEEN, M. G., LANGLEY, D. J., AND KORT, J. Three approaches to take the user perspective into account during new

product design. *International Journal of Innovation Management 12*, 03 (2008), 275–294.

[165] PARRY, K. W. Grounded theory and social process: A new direction for leadership research. *The Leadership Quarterly 9*, 1 (1998), 85–105.

[166] PATTON, M. Q. *Qualitative research & evaluation methods*. Sage Publications, Incorporated, 2001.

[167] PIKKARAINEN, M., HAIKARA, J., SALO, O., ABRAHAMSSON, P., AND STILL, J. The impact of agile practices on communication in software development. *Empirical Software Engineering 13*, 3 (2008), 303–337.

[168] POPPENDIECK, M., AND POPPENDIECK, T. *Lean software development: An agile toolkit*. Addison-Wesley Professional, 2003.

[169] PRIES-HEJE, L., AND PRIES-HEJE, J. Why scrum works: A case study from an agile distributed project in denmark and india. In *Agile Conference (AGILE), 2011* (2011), IEEE, pp. 20–28.

[170] QUINN, R. W., AND DUTTON, J. E. Coordination as energy-in-conversation. *Academy of management review 30*, 1 (2005), 36–57.

[171] REDPATH, L., HURST, D., AND DEVINE, K. Knowledge workers, managers, and contingent employment relationships. *Personnel review 38*, 1 (2008), 74–89.

[172] REJAB, M. M., NOBLE, J., AND ALLAN, G. Distributing expertise in agile software development projects. In *Agile Conference (AGILE), 2014* (2014), IEEE, pp. 33–36.

[173] REJAB, M. M., NOBLE, J., AND ALLAN, G. Locating expertise in agile software development projects. In *Agile Processes in Software Engineering and Extreme Programming*. Springer, 2014, pp. 260–268.

[174] REN, Y., KIESLER, S., AND FUSSELL, S. R. Multiple group coordination in complex and dynamic task environments: Interruptions, coping mechanisms, and technology recommendations. *Journal of management information systems 25*, 1 (2008), 105–130.

[175] REYCHAV, I., AND TEENI, D. Knowledge exchange in the shrines of knowledge: The hows and wheres of knowledge sharing processes. *Computers & Education 53*, 4 (2009), 1266–1277.

[176] RICO, R., SÁNCHEZ-MANZANARES, M., GIL, F., AND GIBSON, C. Team implicit coordination processes: A team knowledge–based approach. *Academy of Management Review 33*, 1 (2008), 163–184.

[177] ROCHE, G., AND VASQUEZ-MCCALL, B. The amazing team race a team based agile adoption. In *Agile Conference, 2009. AGILE'09.* (2009), IEEE, pp. 141–146.

[178] RUSSELL, D. L., AND GOODNIGHT, J. E. The rolling learning cell: a method to integrate individual assessment and team grading components in information systems curriculum team projects. *Information Systems Education Journal 7*, 37 (2009), 1–15.

[179] RYAN, S., AND O'CONNOR, R. V. Development of a team measure for tacit knowledge in software development teams. vol. 82, Elsevier, pp. 229–240.

[180] RYAN, S., AND O'CONNOR, R. V. *Social interaction, team tacit knowledge and transactive memory: empirical support for the agile approach*. 2012.

[181] SANTOS, V., GOLDMAN, A., AND DE SOUZA, C. R. Fostering effective inter-team knowledge sharing in agile software development. *Empirical Software Engineering* (2014), 1–46.

[182] SANTOS, V., GOLDMAN, A., AND RORIZ FILHO, H. The influence of practices adopted by agile coaching and training to foster interaction and knowledge sharing in organizational practices. In *System Sciences (HICSS), 2013 46th Hawaii International Conference on* (2013), IEEE, pp. 4852–4861.

[183] SANTOS, V. A., GOLDMAN, A., AND SANTOS, C. D. Uncovering steady advances for an extreme programming course. *CLEI Electronic Journal 15*, 1 (2012), 2–2.

[184] SANTOS, V. A., GOLDMAN, A., AND SANTOS, C. D. Uncovering steady advances for an extreme programming course. vol. 15. Centro Latinoamericano de Estudios en Informtica, 2012, pp. 2–2.

[185] SARMA, A., VAN DER HOEK, A., AND REDMILES, D. *The coordination pyramid: A perspective on the state of the art in coordination technology*. IEEE, 2010.

[186] SCARDAMALIA, M., AND BEREITER, C. Literate expertise. *Toward a general theory of expertise: Prospects and limits* (1991), 172–194.

[187] SCHWABER, K., AND BEEDLE, M. *Agile software development with Scrum*, vol. 1. Prentice Hall Upper Saddle River, 2002.

[188] SCOTT, E., AND POLLOCK, M. Effectiveness of self-selected teams: A systems development project experience. *Issues in Informing Science and Information Technology 3* (2006), 601–617.

[189] SEAMAN, C. B. Qualitative methods in empirical studies of software engineering. *Software Engineering, IEEE Transactions on 25*, 4 (1999), 557–572.

[190] SHAKIR, R. Soft skills at the malaysian institutes of higher learning. *Asia Pacific Education Review 10*, 3 (2009), 309–315.

[191] SHANKARMANI, R., MANTHA, S., AND BABU, V. Performance assessment of ASD team using FPL football rules as reference. In *India Conference (INDICON), 2011 Annual IEEE* (2011), IEEE, pp. 1–4.

[192] SHANTEAU, J., WEISS, D. J., THOMAS, R. P., AND POUNDS, J. C. Performance-based assessment of expertise: How to decide if someone is an expert or not. *European Journal of Operational Research 136*, 2 (2002), 253–263.

[193] SHARP, H., AND ROBINSON, H. An ethnographic study of xp practice. *Empirical Software Engineering 9*, 4 (2004), 353–375.

[194] SHARP, H., AND ROBINSON, H. *Three 'C's of agile practice: collaboration, coordination and communication*. Dingsyr, Torgeir and Dyb, Tore and Moe, Nils Brede, Agile Software Development: Current Research and Future Directions, Springer Publishing Company, Incorporated, 2010.

[195] SHIM, J., SHEU, T., CHEN, H., JIANG, J., AND KLEIN, G. Coproduction in successful software development projects. *Information and Software Technology 52*, 10 (2010), 1062–1068.

[196] SILVA, K., AND DOSS, C. The growth of an agile coach community at a fortune 200 company. In *Agile Conference (AGILE), 2007* (2007), IEEE, pp. 225–228.

[197] SOMMERVILLE, I. Software engineering. International computer science series. *ed: Addison Wesley* (2004).

[198] SRINIVASAN, J., AND LUNDQVIST, K. Agile in india: Challenges and lessons learned. In *Proceedings of the 3rd India software engineering conference* (2010), ACM, pp. 125–130.

[199] STAPLETON, J. *DSDM Dynamic Systems Development Method: the method in practice*. Cambridge University Press, 1997.

[200] STERN, P. N., AND PORR, C. *Essentials of accessible grounded theory*. Left Coast Press, 2010.

[201] STETTINA, C. J., AND HEIJSTEK, W. Necessary and neglected?: an empirical study of internal documentation in agile software development teams. In *Proceedings of the 29th ACM international conference on Design of communication* (2011), ACM, pp. 159–166.

[202] STOL, K.-J., RALPH, P., AND FITZGERALD, B. Grounded theory in software engineering research: a critical review and guidelines. In *Proceedings of the 38th International Conference on Software Engineering* (2016), ACM, pp. 120–131.

[203] STRAUSS, A., AND CORBIN, J. *Basics of qualitative research: Procedures and techniques for developing grounded theory*. Thousand Oaks, CA: Sage, 1998.

[204] STRODE, D., HUFF, S., HOPE, B., AND LINK, S. Coordination in co-located agile software development projects. *Journal of Systems and Software* (2012).

[205] STRODE, D. E., AND HUFF, S. L. A taxonomy of dependencies in agile software development. In *ACIS 2012: Location, location, location: Proceedings of the 23rd Australasian Conference on Information Systems 2012* (2012), ACIS, pp. 1–10.

[206] SUDDABY, R. From the editors: What grounded theory is not. *Academy of Management Journal 49*, 4 (2006), 633–642.

[207] SUTHERLAND, J. Scrum handbook. *See URL: http://jeffsutherland.com/scrumhandbook. pdf (12-03-25)* (2010).

[208] SUTHERLAND, J., VIKTOROV, A., AND BLOUNT, J. Adaptive engineering of large software projects with distributed/outsourced teams.

In *Proc. International Conference on Complex Systems, Boston, MA, USA* (2006), pp. 25–30.

[209] SUTHERLAND, J., VIKTOROV, A., BLOUNT, J., AND PUNTIKOV, N. Distributed scrum: Agile project management with outsourced development teams. In *System Sciences, 2007. HICSS 2007. 40th Annual Hawaii International Conference on* (2007), IEEE, pp. 274a–274a.

[210] TAKEUCHI, H., AND NONAKA, I. The new new product development game. *Harvard business review 64*, 1 (1986), 137–146.

[211] TENGSHE, A., AND NOBLE, S. Establishing the agile PMO: Managing variability across projects and portfolios. In *Agile Conference (AGILE), 2007* (2007), IEEE, pp. 188–193.

[212] TRIPP, J. F., AND RIEMENSCHNEIDER, C. K. Toward an understanding of job satisfaction on agile teams: Agile development as work redesign. In *System Sciences (HICSS), 2014 47th Hawaii International Conference on* (2014), IEEE, pp. 3993–4002.

[213] TRIPP, L. L. Benefits of certification. *Computer 35*, 6 (2002), 31–33.

[214] URQUHART, C., LEHMANN, H., AND MYERS, M. D. Putting the 'theory' back into grounded theory: guidelines for grounded theory studies in information systems. *Information Systems Journal 20*, 4 (2009), 357–381.

[215] URQUHART, C., LEHMANN, H., AND MYERS, M. D. Putting the theoryback into grounded theory: guidelines for grounded theory studies in information systems. *Information systems journal 20*, 4 (2010), 357–381.

[216] VAN DE VEN, A. H., DELBECQ, A. L., AND KOENIG JR, R. Determinants of coordination modes within organizations. *American sociological review* (1976), 322–338.

[217] VANHANEN, J., AND KORPI, H. Experiences of using pair programming in an agile project. In *System Sciences, 2007. HICSS 2007. 40th Annual Hawaii International Conference on* (2007), IEEE, pp. 274b–274b.

[218] VERBURG, R. M., AND ANDRIESSEN, E. J. A typology of knowledge sharing networks in practice. *Knowledge and Process Management 18*, 1 (2011), 34–44.

[219] VERSIONONE. 7th Anuual State of Agile Dev Survey. `http://www.versionone.com/state-of-agile-survey-results/`, 2012. [Online; accessed 19-March-2013].

[220] VINEKAR, V., SLINKMAN, C. W., AND NERUR, S. Can agile and traditional systems development approaches coexist? an ambidextrous view. *Information systems management 23*, 3 (2006), 31–42.

[221] VISCHER, J. C. Towards an environmental psychology of workspace: how people are affected by environments for work. *Architectural Science Review 51*, 2 (2008), 97–108.

[222] WANG, X., O'CONCHUIR, E., AND VIDGEN, R. *A paradoxical perspective on contradictions in agile software development*. 2008.

[223] WANG, Y., SANG, D., AND XIE, W. Analysis on agile software development methods from the view of informationalization supply chain management. In *Intelligent Information Technology Application Workshops, 2009. IITAW'09. Third International Symposium on* (2009), IEEE, pp. 219–222.

[224] WATERMAN, M., NOBLE, J., AND ALLAN, G. How much up-front? a grounded theory of agile architecture. In *Software Engineering (ICSE), 2015 IEEE/ACM 37th IEEE International Conference on* (2015), vol. 1, IEEE, pp. 347–357.

[225] WEBER, R. Evaluating and developing theories in the information systems discipline. *Journal of the Association for Information Systems 13*, 1 (2012), 1–30.

[226] WEGNER, D. M. Transactive memory: A contemporary analysis of the group mind. *Theories of group behavior 185* (1987), 208.

[227] WEGNER, D. M. A computer network model of human transactive memory. *social cognition 13*, 3 (1995), 319–339.

[228] WEICK, K. E., AND ROBERTS, K. H. Collective mind in organizations: Heedful interrelating on flight decks. *Administrative science quarterly* (1993), 357–381.

[229] WEIMAR, E., NUGROHO, A., VISSER, J., AND PLAAT, A. Towards high performance software teamwork. In *Proceedings of the 17th International Conference on Evaluation and Assessment in Software Engineering* (2013), ACM, pp. 212–215.

[230] WHETTEN, D. A. What constitutes a theoretical contribution? *Academy of management review 14*, 4 (1989), 490–495.

[231] WHITAKER, J., MITHAS, S., AND KRISHNAN, M. S. Organizational learning and capabilities for onshore and offshore business process outsourcing. *Journal of Management Information Systems 27*, 3 (2010), 11–42.

[232] WHITWORTH, E., AND BIDDLE, R. The social nature of agile teams. In *Agile Conference (AGILE), 2007* (2007), IEEE, pp. 26–36.

[233] YU, X., AND PETTER, S. Understanding agile software development practices using shared mental models theory. *Information and Software Technology 56*, 8 (2014), 911–921.

A

Human Ethics Approval for Interviews

The following documents are approval from the Human Ethics Committee (HEC) of Victoria University of Wellington and supporting documents for conducting interviews of this study.

TE WHARE WĀNANGA O TE ŪPOKO O TE IKA A MĀUI
VICTORIA
UNIVERSITY OF WELLINGTON

Phone 0-4-463 5676
Fax 0-4-463 5209
Email Allison.kirkman@vuw.ac.nz

MEMORANDUM

TO	Mawarny Md Rejab
COPY TO	James Noble George Allan
FROM	Dr Allison Kirkman, Convener, Human Ethics Committee
DATE	3 September 2012
PAGES	1
SUBJECT	**Ethics Approval: 19488** **Expertise Coordination in Agile Software Development Projects**

Thank you for your application for ethical approval, which has now been considered by the Standing Committee of the Human Ethics Committee.

Your application has been approved from the above date and this approval continues until 30 June 2015. If your data collection is not completed by this date you should apply to the Human Ethics Committee for an extension to this approval.

Best wishes with the research.

Allison Kirkman
Human Ethics Committee

Appendix D

HUMAN ETHICS COMMITTEE

Application for Approval of Research Projects

Please write legibly or type if possible. Applications must be signed by supervisor (for student projects) and Head of School

Note: The Human Ethics Committee attempts to have all applications approved within three weeks but a longer period may be necessary if applications require substantial revision.

1. NATURE OF PROPOSED RESEARCH:

a. ~~Staff Research~~/ **Student Research** (delete one)

b. If Student Research ……..... Degree : **PhD (Software Engineering)**

Course Code : **SWEN690**

c. Project Title: **Expertise Coordination in Agile Software Development Projects**

2. INVESTIGATORS:

(a) Principal Investigator

Name :**Mawarny Md. Rejab**

Email address :**Mawarny.Md.Rejab@ecs.vuw.ac.nz**

School/Dept/Group :**Engineering & Computer Science/ Software Engineering**

(b) Other Researchers Name Position

None

(c) Supervisor (in the case of student research projects)

Prof James Noble

Dr George Allan

3. DURATION OF RESEARCH:

(a) Proposed starting date for data collection:

10 September 2012

(**Note:** that NO part of the research requiring ethical approval may commence prior to approval being given)

(b) Proposed date of completion of project as a whole: **30 June 2015**

4. PROPOSED SOURCE/S OF FUNDING AND OTHER ETHICAL CONSIDERATIONS

(a) Sources of funding for the project

Please indicate any ethical issues or conflicts of interest that may arise because of sources of funding e.g. restrictions on publication of results

None

(b)Is any professional code of ethics to be followed: Y ☐ N ■

If yes, name

(c) Is ethical approval required from any other body Y ☐ N ■

If yes, name and indicate when/if approval will be given

5. DETAILS OF PROJECT

Briefly Outline:

(a) The objectives of the project:

a) To explore and understand the current issues of expertise coordination in Agile Software Development projects.

b) To determine the factors which influence processes of expertise coordination in Agile Software Development projects.

c) To explore and understand how expertise is coordinated in Agile Software Development projects which will lead to forming an appropriate expertise coordination framework.

(b) Method of data collection

This study will be carried out using qualitative research method including Grounded Theory. This study requires data gathering from various Agile Software Development projects which are run in New Zealand and possibly Australia and the USA in order to ensure the reliability and validity of data. Data collection will be by a semi-structured interview technique. A set of open-ended questions will be prepared before the interview and will be answered by voluntary individual participant. Each interview session would take one hour and will be held at the interviewee's workplace or elsewhere with mutual agreement between researcher and interviewee. No personal details such as participant's name or company's name will be collected during the interview session. All findings collected will be kept confidential and will be destroyed after the completion of the research.

(c)The benefits and scientific value of the project

a) **This study will provide evidence that will help researchers gain an in-depth understanding of the expertise issues and coordination in Agile Software Development. As such, it will contribute to the body of knowledge in the fields of software engineering by inferring new working hypotheses leading to new grounded theory.**

b) **This should provide procedures for managing the expertise resources in Agile Software Development teams.**

c) **The outcome of this study could provide comprehensive requirements and procedures for coordinating expertise resources particularly in the circulation of expertise among Agile team members.**

(d) Characteristics of the participants

This study will involve software developers, team leaders, business analysts, and software designers who are currently engaged in Agile Software Development projects. The participants of this study should take part voluntarily.

(e) Method of recruitment

A list of companies who are engaged in Agile Software Development projects and Agile Practitioners will be identified via Agile community lists or user groups such as Agile Professional Networks (APN). They will be provided with a brief outline of the intended research via email. The researcher will collaborate with each interested organisation to develop mutually agreeable terms for their participation in the research. The researcher will elicit their participation on a voluntary basis and all collected data will be kept confidentially. A document which contains the purpose and scope of this research, the time commitments expected from the participants, the processes of data collection, the use of data and other commitments expected from participants will be made available to the participants. The consent document, information sheet and interview guide are attached to this application.

(f)Payments that are to be made/expenses to be reimbursed to participants

None

(g) Other assistance (e.g. meals, transport) that is to be given to participants

None

(h) Any special hazards and/or inconvenience (including deception) that participants will encounter

None

(i) State whether consent is for (delete where not applicable):

(i) **the collection of data**

(ii) ~~attribution of opinions or information~~

(iii) ~~release of data to others~~

(iv) **use for a conference report or a publication**

(iv) **use for a PhD thesis**

Attach a copy of any questionnaire or interview schedule to the application: **attached**

(j)How is informed consent to be obtained (see sections 4.1, 4.5(d) and 4.8(g) of the Human Ethics Policy)

(i) the research is strictly anonymous, an information sheet is supplied and informed consent is implied by voluntary participation in filling out a questionnaire for example (include a copy of the information sheet) Y ☐ N ■

(ii) the research is not anonymous but is confidential and informed consent will be obtained through a signed consent form (include a copy of the consent form and information sheet) Y ■ N ☐

(iii) the research is neither anonymous or confidential and informed consent will be obtained through a signed consent form (include a copy of the consent form and information sheet) Y ☐ N ■

(iv) informed consent will be obtained by some other method (please specify and provide details) Y ☐ N ■

With the exception of anonymous research as in (i), if it is proposed that written consent will not be obtained, please explain why

Not applicable

(k) If the research will not be conducted on a strictly anonymous basis state how issues of confidentiality of participants are to be ensured if this is intended. (See section 4..1(e) of the Human Ethics Policy). (E.g. who will listen to tapes, see questionnaires or have access to data). Please ensure that you distinguish clearly between anonymity and confidentiality. Indicate which of these are applicable.

(i) access to the research data will be restricted to the investigator Y ☐ N ■

(ii) access to the research data will be restricted to the investigator and their supervisor (student research) Y ■ N ☐

(iii) all opinions and data will be reported in aggregated form in such a way that individual persons or organisations are not identifiable Y ■ N ☐

(iv) Other (please specify)

Not Applicable

(l)Procedure for the storage of, access to and disposal of data, both during and at the conclusion of the research. (see section 4.12 of the Human Ethics Policy). Indicate which are applicable:

(i) all written material (questionnaires, interview notes, etc) will be kept in a locked file and access is restricted to the investigator Y ■ N ☐

(ii) all electronic information will be kept in a password-protected file and access will be restricted to the investigator — Y ■ N ☐

(iii) all questionnaires, interview notes and similar materials will be destroyed:

(a) at the conclusion of the research — Y ☐ N ■

(b)**Four (4)** years after the conclusion of the research; or — Y ■ N ☐

(iv) any audio or video recordings will be returned to participants and/or electronically wiped — Y ■ N ☐

(v) other procedures (please specify):

Access will be restricted to the investigator and supervisors

If data and material are not to be destroyed please indicate why and the procedures envisaged for ongoing storage and security

Not Applicable

(m) Feedback procedures (See section 7 of Appendix 1 of the Human Ethics Policy). You should indicate whether feedback will be provided to participants and in what form. If feedback will not be given, indicate the reasons why.

The participants will be continuously updated about the research and all essential findings, and provided with prompt feedback to confirm or review the researcher's interpretation of given data.

(n) Reporting and publication of results. Please indicate which of the following are appropriate. The proposed form of publications should be indicated on the information sheet and/or consent form.

(i) publication in academic or professional journals — Y ■ N ☐

(ii) dissemination at academic or professional conferences — Y ■ N ☐

(iii) deposit of the research paper or thesis in the University Library (student research) — Y ■ N ☐

(iv) other (please specify)

None

Signature of investigators as listed on page 1 **(including supervisors) and Head of School**.

NB: All investigators and the Head of School must sign before an application is submitted for approval

……

Date……...

……

Date……...

……

Date……...

Head of School:

……

Date……...

APPLICATIONS FOR HUMAN ETHICS APPROVAL

CHECKLIST

- Have you read the Human Ethics Policy?
- Is ethical approval required for your project?
- Have you established whether informed consent needs to be obtained for your project?
- In the case of student projects, have you consulted your supervisor about any human ethics implications of your research?
- Has your supervisor read and signed the application?
- Have you included an information sheet for participants which explains the nature and purpose of your research, the proposed use of the material collected, who will have access to it, whether the data will be kept confidential to you, how anonymity or confidentiality is to be guaranteed?
- Have you included a written consent form?
- If not, have you explained on the application form why you do not need to get written consent?
- Are you asking participants to give consent to:
    - collect data from them
    - attribute information to them
    - release that information to others
    - use the data for particular purposes
- Have you indicated clearly to participants on the information sheet or consent form how they will be able to get feedback on the research from you (e.g. they may tick a box on the consent form indicating that they would like to be sent a summary), and how the data will be stored or disposed of at the conclusion of the research?
- Have you included a copy of any questionnaire or interview checklist you propose using?
- Has your application been seen by the head of your school or department (or the person given responsibility to consider applications on behalf of the head (see section 4.5(b) of the Human Ethics Policy).

PLEASE FORWARD YOUR COMPLETED APPLICATION FORM TO THE SECRETARY, HUMAN ETHICS COMMITTEE OR, IN THE CASE OF APPLICATIONS FROM SCHOOLS OR DEPARTMENTS WITH AN APPROVED ETHICS SUB-COMMITTEE, TO THE CONVENER OF THAT SUB-COMMITTEE

VICTORIA UNIVERSITY OF WELLINGTON
Te Whare Wananga o te Upoko o te Ika a Maui

Attachment A: Cover Letter for Research Participant

[Participant's name]
[Company name]
[Employment address]

Dear [Participant's name],

PhD Research into the Expertise Coordination for Agile Software Development Project

Thank you for your interest in being interviewed for this research.

Prior to conducting the interview, Victoria University of Wellington requires that I obtain your written informed consent. This consent is a normal part of any research project and forms one criterion of the Victoria University Human Ethics Committee Guidelines that I must meet.

Attached herewith are:

- A copy of the research document that outlines the purpose, scope & approach to the project.
- An interview guide that describes the topics that will be discussed during the interview session.
- A consent form that you need to sign and return to me [in the enclosed envelope] if you decide to participate in this research.

I would like to emphasise that you do not have to participate in this research and that you are free to withdraw from this research without explanation up to one month after the interview.

If you require any further information or if you would prefer not to be interviewed, please feel free to email me at Mawarny.Md.Rejab@ecs.vuw.ac.nz or call me on +02230034839. Alternatively you may contact either of my supervisors Prof James Noble (kjx@ecs.vuw.ac.nz or +64 4 463 6736) or Dr. George Allan (george.allan@ecs.vuw.ac.nz or +64 4 4636741)

Thank you.

Yours sincerely

Mawarny Md. Rejab
(PhD Student)
Mobile: +02230034839
E-mail:Mawarny.Md.Rejab@ecs.vuw.ac.nz

VICTORIA UNIVERSITY OF WELLINGTON
Te Whare Wananga o te Upoko o te Ika a Maui

Attachment B: Information Sheet

General Information
This research study contributes towards the requirements for a PhD degree at Victoria University of Wellington, New Zealand.

Research Topic: **Expertise Coordination in Agile Software Development Project**

PhD Student: Mawarny Md. Rejab (Mawarny.Md.Rejab@ecs.vuw.ac.nz, 0223034839)
Supervisors: Prof. James Noble (kjx@ecs.vuw.ac.nz, +64 4 4636736)
Dr. George Allan (george.allan@ecs.vuw.ac.nz, +64 4 463 6741)

Objectives of Research
The objective of this research is to explore and understand the current issues of expertise coordination in an Agile Software Development project. This research also aims to determine the factors which influence processes of expertise coordination in an Agile Software Development project. The core of this research is to explore and understand how expertise is coordinated in Agile Software Development which will lead to the creation of an appropriate expertise coordination framework.

Research Methodology
This research intends to use a qualitative research methodology by selecting Grounded Theory technique. The grounded theory is selected due to this research involves the exploration of human interaction in Agile Software Development project. We have sought and have been granted approval has been granted by the University's Human Ethics Committee to conduct semi-structured interviews.

The researcher aims to gather valuable data regarding expertise coordination issues by interviewing project managers, software developers, software team leaders, business analyst, agile practitioners and consultants who are involved in Agile Software Development project. The interviews will be conducted by research with voluntary individual participant. Each interview session would take one hour and will be held at the interviewee's workplace or any place depends on the mutual agreement between researcher and interviewee. An interview agenda has been attached to this information sheet outlining what data will be sought during the interviews.

Purpose of Data Collection
The data gathered will be analysed to derive factors and processes of expertise coordination in Agile Software Development project. Findings from the research will be published in conferences papers and referred journals for the benefit of the larger research community. The final PhD thesis will be published and can be accessed at Victoria University library of Wellington, New Zealand.

VICTORIA UNIVERSITY OF WELLINGTON
Te Whare Wananga o te Upoko o te Ika a Maui

Confidentiality and Consent

The data collected will be kept in a confidential way and will be destroyed at the completion of the research. The participant's details including company background will not be recorded during the interview. All data collected for this research will be remained confidential to the researcher, Mawarny Md. Rejab and her supervisors, Prof James Noble and Dr. George Allan.

VICTORIA UNIVERSITY OF WELLINGTON
Te Whare Wananga o te Upoko o te Ika a Maui

Attachment C: Consent for Research Participant

Research Title: Expertise Coordination in Agile Software Development Teams
Researcher : Mawarny Md. Rejab

I have been given and have understood an explanation of this research project and the confidentiality conditions. I have had an opportunity to ask questions and have them answered to my satisfaction.

I agree to be interviewed by Mawarny Md. Rejab for the purpose of this research for her PhD degree and resultant theses and publications. I consent to the collection and use of my perceptions, experiences, opinions and information in this research.

I understand that I may withdraw from this research up to the one month after the interview without penalty or explanation. This can be done by contacting the researcher or on her supervisors.

Do you agree to have interviews sound-recorded?

☐ Yes ☐ No

Would you like to receive a copy of any publications that are based on these interviews?

☐ Yes ☐ No

If yes, please provide an email address, or mailing address that we can use to send a copy of the publication.

_____

_____

Name: _____

Signed: _____

Date:_____

VICTORIA UNIVERSITY OF WELLINGTON
Te Whare Wananga o te Upoko o te Ika a Maui

Attachment D: Interview Guide

Interview Details:

Interview Date : _____

Interview Time :_____(1 hour)

Interview Venue :_____

Topic: Expertise Coordination for Agile Software Development Projects

Objectives:

- To gain in-depth understanding of the expertise issues in Agile Software Development projects.
- To understand how to manage the expertise resources and its dependencies in Agile Software Development projects.
- To understand how Agile teams rely on expertise outside Agile teams.

Agenda:

No	Activities	Duration (minutes)
1	Both parties agree with the defined outcomes, agenda and rules for the interview	5
2	Interviewee defines roles, responsibilities, skills, and experiences in Agile Software Development projects.	10
3	Discuss the current Agile Software Development practices that lead to expertise coordination.	10
4	Discuss how to support expertise coordination in Agile Software Development teams.	10
5	Discuss issues or difficulties that Agile teams face in managing expertise resources and its dependencies.	10
6	Suggest improvements or recommendations to enable expertise coordination in the current Agile Software Development practices.	10
7	Wrap-up (follow up session and feedback)	5
	Total	60

Semi-structured Interview Questions Guidelines:

1) What are your roles and responsibilities in the Agile Software Development project?
2) Which Agile Software Development methodology has been used in the software development project?
3) Tell me about your cross-functional team?
4) Could you explain how do you improve your skills and knowledge to become as a T-shaped person in your Agile team?
5) Do your team members recognize and trust your capability and expertise? Why and why not?
6) Do you rely on other team members in developing your expertise? How?
7) How often have you been referred to by your team members when they face a problem in solving tasks during software development? What are factors that influence them to seek for your assistance?
8) What do you think about sharing your expertise with your team members? How do you share your expertise?
9) What challenges or barriers did you face during sharing your expertise with your team members? What are the strategies or actions that you took to overcome these challenges?
10) How do you identify expertise among your team members?
11) What do you think about seeking for assistance or help from your team members when you face problems in solving tasks during software development?
12) Please elaborate on the process of seeking help from other expertise in your team.
13) What challenges or barriers did you face in seeking expertise in your software development team? What are strategies or actions that you took to overcome these challenges?
14) What Agile practices and artefacts that do you think related to managing expertise and its dependency in Agile teams?
15) Do you deal with external expertise such as DBA and user experience designer in accomplishing your tasks?
16) What challenges or barriers did you face when dealing with external expertise? What are strategies or actions that you took to overcome these challenges?
17) Is there anything else that we should discuss? If yes, please elaborate more.

B

Human Ethics Approval for Observations

The following documents are approval from the Human Ethics Committee (HEC) of Victoria University of Wellington and supporting documents for conducting observations of this study.

TE WHARE WĀNANGA O TE ŪPOKO O TE IKA A MĀUI
VICTORIA
UNIVERSITY OF WELLINGTON

Phone 0-4-463 5676
Fax 0-4-463 5209
Email Allison.kirkman@vuw.ac.nz

MEMORANDUM

TO	Mawarny Md Rejab
COPY TO	James Noble George Allan
FROM	Dr Allison Kirkman, Convener, Human Ethics Committee
DATE	12 September 2013
PAGES	1
SUBJECT	**Ethics Approval: 19488** **Expertise Coordination in Agile Software Development Projects**

Thank you for your request to amend your ethics approval. This has now been considered and the request granted.

Your application has approval until 30 June 2015. If your data collection is not completed by this date you should apply to the Human Ethics Committee for an extension to this approval.

Best wishes with your research.

Allison Kirkman
Human Ethics Committee

Appendix D

TE WHARE WĀNANGA O TE ŪPOKO O TE IKA A MĀUI
VICTORIA
UNIVERSITY OF WELLINGTON

HUMAN ETHICS COMMITTEE

Application for Approval of Research Projects

Please write legibly or type if possible. Applications must be signed by supervisor (for student projects) and Head of School

Note: The Human Ethics Committee attempts to have all applications approved within three weeks but a longer period may be necessary if applications require substantial revision.

1. NATURE OF PROPOSED RESEARCH:

a. ~~Staff Research~~/ **Student Research** (delete one)

b. If Student Research ……..... Degree : **PhD (Software Engineering)**

Course Code : **SWEN690**

c. Project Title: **Expertise Coordination for Agile Software Development Projects**

2. INVESTIGATORS:

(a) Principal Investigator

Name :**Mawarny Md. Rejab**

Email address :**Mawarny.Md.Rejab@ecs.vuw.ac.nz**

School/Dept/Group :**Engineering & Computer Science/ Software Engineering**

(b) Other Researchers Name Position

None

(c) Supervisor (in the case of student research projects)

Prof James Noble

Dr George Allan

3. DURATION OF RESEARCH:

(a) Proposed starting date for data collection:

30 September 2013

(**Note:** that NO part of the research requiring ethical approval may commence prior to approval being given)

(b) Proposed date of completion of project as a whole: **30 June 2015**

4. PROPOSED SOURCE/S OF FUNDING AND OTHER ETHICAL CONSIDERATIONS

(a) Sources of funding for the project

Please indicate any ethical issues or conflicts of interest that may arise because of sources of funding e.g. restrictions on publication of results

None

(b)Is any professional code of ethics to be followed: Y ☐ N ■

If yes, name

(c) Is ethical approval required from any other body Y ☐ N ■

If yes, name and indicate when/if approval will be given

5. DETAILS OF PROJECT

Briefly Outline:

(a) The objectives of the project:

The aim of this study is to build a theory of expertise coordination in Agile software development projects, which is guided by three research questions:

- **How is expertise coordinated in Agile software development projects?**
  This question attempts to understand the expertise coordination processes that should be carried out by Agile team members in managing expertise dependencies.
- **What factors influence expertise coordination in Agile software development projects?**
  This question attempts to identify factors that should be considered in coordinating expertise in Agile teams.
- **What expertise coordination practices should be applied in Agile software development projects?**

This question attempts to determine practices that Agile team members should apply in coordinating expertise in Agile teams.

(b) Method of data collection

This study is carried out by employing Grounded Theory. This study employs interviews as the predominant source of data collection (Ethics Approval: 19488). However, multiple sources of data will strengthen the Grounded Theory research. Therefore, the researcher intends to include participant observation in conjunction with the interview.

Participant observation provides a great opportunity to view the actual participants' behaviour when they are engaged in the software project development. Participant observation enables the researcher to gain a deeper understanding of the participants' settings. As a secondary data collection method, participant observation also will allow the researcher to confirm the accuracy of interviews and enhance the validity of data.

The potential participants for this study are Agile practitioners who engage in Agile software development projects. Thus, participant observation requires the researcher to identify software companies that apply Agile practices in their software development projects. Moderate participant involvement will be used during the observation by sharing work experiences together. The researcher will keep written records and take some pictures during the observation. No personal details such as participant's name or company's name will be collected during the interview session. All findings will be kept confidential and will be destroyed after the completion of the research.

(c)The benefits and scientific value of the project

The planned contributions of this study can be summarized into theoretical, functional, and practical contributions as follows:

- **This study will contribute theories on expertise coordination in Agile software development projects. The novelty of this study lies in the demonstration of theories related to the processes of expertise coordination, factors that influence expertise coordination, and expertise coordination practices in Agile teams.**

- **This study will provide empirical evidence that will assist Agile practitioners and researchers to gain in depth understanding of the expertise coordination in Agile Software Development projects. This evidence is important to contribute to the body of knowledge in software engineering and can serve as working hypotheses that could be tested by future researchers.**
- **This study will provide guidelines on how to coordinate expertise in Agile software development projects. The guidelines will also assist Agile practitioners in managing expertise resources and circulating the available expertise effectively. This guideline also will assist Agile teams to coordinate expertise and rely on roles outside the teams.**

(d) Characteristics of the participants

The participants for this study are Agile practitioners who engage in Agile software development projects. Eligibility criteria should be considered in recruiting participants for this study including Agile methods used and roles in Agile teams.

- **Agile methods**

  There is no restriction as to which Agile methods have been used. This study is open to recruit Agile practitioners even though there is no specific Agile method is used in their software projects.

- **Agile roles**

  More priority is put on Agile roles since this study requires a broad range of roles. The wide variety of roles is important to enable the triangulation of findings. Different roles will provide different insights and perspectives on expertise coordination in Agile teams.

The participants of this study should take part voluntarily.

(e) Method of recruitment

Agile Professional Network (APN) is a avenue to identify Agile practitioners from a different software organizations based in New Zealand. Attending Agile seminar and workshops which are organized by APN or software companies is another way to recruit more participants. Additional to APN, Agile conferences

which run around the world can be used as a platform to recruit a wide range of participants.

The potential participants will be provided with a brief outline of the intended research via email. The researcher will contact the interested software companies to develop mutually agreeable terms for their participation in the research. The researcher will elicit their participation on a voluntary basis and all collected data will be kept confidentially.

Before conducting the observation, the participants will be provided with a document that consists of the purpose and scope of this study, the time commitments expected from the participants, the processes of data collection, and other commitments expected from participants. The consent document, information sheet and observation guide are attached together to this application.

(f)Payments that are to be made/expenses to be reimbursed to participants

None

(g) Other assistance (e.g. meals, transport) that is to be given to participants

None

(h) Any special hazards and/or inconvenience (including deception) that participants will encounter

None

(i) State whether consent is for (delete where not applicable):

(i) **the collection of data**

(ii) ~~attribution of opinions or information~~

(iii) ~~release of data to others~~

(iv) **use for a conference report or a publication**

(iv) **use for a PhD thesis**

Attach a copy of any questionnaire or interview schedule to the application: **attached**

(j)How is informed consent to be obtained (see sections 4.1, 4.5(d) and 4.8(g) of the Human Ethics Policy)

(i) the research is strictly anonymous, an information sheet is supplied and informed consent is implied by voluntary participation in filling out a questionnaire for example (include a copy of the information sheet) Y ☐ N ■

(ii) the research is not anonymous but is confidential and informed consent will be obtained through a signed consent form (include a copy of the consent form and information sheet) Y ■ N ☐

(iii) the research is neither anonymous or confidential and informed consent will be obtained through a signed consent form (include a copy of the consent form and information sheet) Y ☐ N ■

(iv) informed consent will be obtained by some other method (please specify and provide details) Y ☐ N ■

With the exception of anonymous research as in (i), if it is proposed that written consent will not be obtained, please explain why

Not applicable

(k) If the research will not be conducted on a strictly anonymous basis state how issues of confidentiality of participants are to be ensured if this is intended. (See section 4..1(e) of the Human Ethics Policy). (E.g. who will listen to tapes, see questionnaires or have access to data). Please ensure that you distinguish clearly between anonymity and confidentiality. Indicate which of these are applicable.

(i) access to the research data will be restricted to the investigator Y ☐ N ■

(ii) access to the research data will be restricted to the investigator and their supervisor (student research) Y ■ N ☐

(iii) all opinions and data will be reported in aggregated form in such a way that individual persons or organisations are not identifiable Y ■ N ☐

(iv) Other (please specify)

Not Applicable

(l)Procedure for the storage of, access to and disposal of data, both during and at the conclusion of the research. (see section 4.12 of the Human Ethics Policy). Indicate which are applicable:

(i) all written material (questionnaires, interview notes, etc) will be kept in a locked file and access is restricted to the investigator Y ■ N ☐

(ii) all electronic information will be kept in a password-protected file and access will be restricted to the investigator — Y ■ N ☐

(iii) all questionnaires, interview notes and similar materials will be destroyed:

(a) at the conclusion of the research — Y ☐ N ■

(b)**Four (4)** years after the conclusion of the research; or — Y ■ N ☐

(iv) any audio or video recordings will be returned to participants and/or electronically wiped — Y ■ N ☐

(v) other procedures (please specify):

Access will be restricted to the investigator and supervisors

If data and material are not to be destroyed please indicate why and the procedures envisaged for ongoing storage and security

Not Applicable

(m) Feedback procedures (See section 7 of Appendix 1 of the Human Ethics Policy). You should indicate whether feedback will be provided to participants and in what form. If feedback will not be given, indicate the reasons why.

The participants will be continuously updated about the research and all essential findings, and provided with prompt feedback to confirm or review the researcher's interpretation of given data.

(n) Reporting and publication of results. Please indicate which of the following are appropriate. The proposed form of publications should be indicated on the information sheet and/or consent form.

(i) publication in academic or professional journals — Y ■ N ☐

(ii) dissemination at academic or professional conferences — Y ■ N ☐

(iii) deposit of the research paper or thesis in the University Library (student research) — Y ■ N ☐

(iv) other (please specify)

None

Signature of investigators as listed on page 1 **(including supervisors) and Head of School**.

NB: All investigators and the Head of School must sign before an application is submitted for approval

……

Date……..……...

……

Date……..……...

……

Date……..……...

Head of School:

……

Date……..……...

APPLICATIONS FOR HUMAN ETHICS APPROVAL

CHECKLIST

- Have you read the Human Ethics Policy?
- Is ethical approval required for your project?
- Have you established whether informed consent needs to be obtained for your project?
- In the case of student projects, have you consulted your supervisor about any human ethics implications of your research?
- Has your supervisor read and signed the application?
- Have you included an information sheet for participants which explains the nature and purpose of your research, the proposed use of the material collected, who will have access to it, whether the data will be kept confidential to you, how anonymity or confidentiality is to be guaranteed?
- Have you included a written consent form?
- If not, have you explained on the application form why you do not need to get written consent?
- Are you asking participants to give consent to:
    - collect data from them
    - attribute information to them
    - release that information to others
    - use the data for particular purposes
- Have you indicated clearly to participants on the information sheet or consent form how they will be able to get feedback on the research from you (e.g. they may tick a box on the consent form indicating that they would like to be sent a summary), and how the data will be stored or disposed of at the conclusion of the research?
- Have you included a copy of any questionnaire or interview checklist you propose using?
- Has your application been seen by the head of your school or department (or the person given responsibility to consider applications on behalf of the head (see section 4.5(b) of the Human Ethics Policy).

PLEASE FORWARD YOUR COMPLETED APPLICATION FORM TO THE SECRETARY, HUMAN ETHICS COMMITTEE OR, IN THE CASE OF APPLICATIONS FROM SCHOOLS OR DEPARTMENTS WITH AN APPROVED ETHICS SUB-COMMITTEE, TO THE CONVENER OF THAT SUB-COMMITTEE

VICTORIA UNIVERSITY OF WELLINGTON
Te Whare Wananga o te Upoko o te Ika a Maui

Attachment A: Cover Letter for Research Participant

[Participant's name]
[Company name]
[Employment address]

Dear [Participant's name],
PhD Research : Expertise Coordination for Agile Software Development Projects

Thank you for your interest to participate in this research.

Prior to conducting the observation, Victoria University of Wellington requires that I obtain your written informed consent. This consent is a normal part of any research project and forms one criterion of the Victoria University Human Ethics Committee Guidelines that I must meet.

Attached herewith are:

- A copy of the research document that outlines the purpose, scope & approach to the project.
- A consent form that you need to sign and return to me [in the enclosed envelope] if you decide to participate in this research.

I would like to emphasise that you do not have to participate in this research and that you are free to withdraw from this research without explanation up to one month after the interview.

If you require any further information or if you would prefer not to be observed, please feel free to email me at Mawarny.Md.Rejab@ecs.vuw.ac.nz or call me on +02230034839. Alternatively you may contact either of my supervisors Prof James Noble (kjx@ecs.vuw.ac.nz or +64 4 463 6736) or Dr. George Allan (george.allan@ecs.vuw.ac.nz or +64 4 4636741)

Thank you.

Yours sincerely

Mawarny Md. Rejab
(PhD Student)
Mobile: +0223034839
E-mail:Mawarny.Md.Rejab@ecs.vuw.ac.nz

VICTORIA UNIVERSITY OF WELLINGTON
Te Whare Wananga o te Upoko o te Ika a Maui

Attachment B: Information Sheet

General Information
This research study contributes towards the requirements for a PhD degree at Victoria University of Wellington, New Zealand.

Research Topic: **Expertise Coordination for Agile Software Development Projects**

PhD Student: Mawarny Md. Rejab (Mawarny.Md.Rejab@ecs.vuw.ac.nz, 0223034839)
Supervisors: Prof. James Noble (kjx@ecs.vuw.ac.nz, +64 4 4636736)
Dr. George Allan (george.allan@ecs.vuw.ac.nz, +64 4 463 6741)

Objectives of Research
The objective of this research is to explore and understand the current issues of expertise coordination in an Agile Software Development project. This research also aims to determine factors which influence processes of expertise coordination in an Agile Software Development project. This research attempts to identify expertise coordination practices that should be carried out by Agile team members to managing their expertise dependencies. The central of this research is to explore and understand how expertise is coordinated in Agile software development projects.

Research Methodology
This research employs Grounded Theory as a qualitative research. The researcher has sought and has been granted approval by the University's Human Ethics Committee to participant observation.

The researcher aims to collect data of expertise coordination in Agile software development projects by using participant observation. The observation will be conducted in the participant's workplace. The potential participants for this study are Agile practitioners who engage in Agile software development projects. Thus, participant observation requires the researcher to identify software companies that apply Agile practices in their software development projects. Moderate participant involvement will be used during the observation by sharing work experiences together. The researcher will keep written records and take some pictures during the observation.

Purpose of Data Collection
The collected data will be analysed to infer theories of expertise coordination for Agile software development projects. Findings from this research will be published in conference papers and referred journals for the benefit of the larger research community. The final PhD thesis will be published and can be accessed at Victoria University library of Wellington, New Zealand.

Confidentiality and Consent

The data collected will be kept confidentially and will be destroyed at the completion of the research. All identification data collected for this research will remain confidential to the researcher, Mawarny Md. Rejab and her supervisors, Prof James Noble and Dr. George Allan.

VICTORIA UNIVERSITY OF WELLINGTON
Te Whare Wananga o te Upoko o te Ika a Maui

Attachment C: Consent for Research Participants

Research Title: Expertise Coordination for Agile Software Development Projects
Researcher : Mawarny Md. Rejab

I have been given and have understood an explanation of this research project and the confidentiality conditions. I have had an opportunity to ask questions and have them answered to my satisfaction.

I agree to participate in this study for Mawarny's PhD degree and resultant theses and publications. I consent to the collection and use of my perceptions, experiences, opinions and information in this research.

I understand that I may withdraw from this research up to the one month after the observation without penalty or explanation. This can be done by contacting the researcher or on her supervisors.

Do you agree to being observed?

☐ Yes ☐ No

Would you like to receive a copy of any publications that are based on the observation results?

☐ Yes ☐ No

If yes, please provide an email address, or mailing address that we can use to send a copy of the publication.

_____

_____

Name: _____

Signed: _____

Date:_____

VICTORIA UNIVERSITY OF WELLINGTON
Te Whare Wananga o te Upoko o te Ika a Maui

C

Human Ethics Approval for Document Analysis

The following documents are approval from the Human Ethics Committee (HEC) of Victoria University of Wellington and supporting documents for undertaking document analysis of this study.

TE WHARE WĀNANGA O TE ŪPOKO O TE IKA A MĀUI
VICTORIA
UNIVERSITY OF WELLINGTON

Phone 0-4-463 5676
Fax 0-4-463 5209
Email Allison.kirkman@vuw.ac.nz

MEMORANDUM

TO	Mawarny Md Rejab
COPY TO	James Noble Stuart Marshall
FROM	Dr Allison Kirkman, Convener, Human Ethics Committee
DATE	30 September 2014
PAGES	1
SUBJECT	**Ethics Approval: 19488** Expertise Coordination in Agile Software Development Projects

Thank you for your request to amend your ethics approval. This has now been considered and the request granted.

Your application has approval until 30 June 2015. If your data collection is not completed by this date you should apply to the Human Ethics Committee for an extension to this approval.

Best wishes with your research.

Allison Kirkman
Human Ethics Committee

Appendix D

TE WHARE WĀNANGA O TE ŪPOKO O TE IKA A MĀUI
VICTORIA
UNIVERSITY OF WELLINGTON

HUMAN ETHICS COMMITTEE

Application for Approval of Research Projects

Please write legibly or type if possible. Applications must be signed by supervisor (for student projects) and Head of School

Note: The Human Ethics Committee attempts to have all applications approved within three weeks but a longer period may be necessary if applications require substantial revision.

~~1.~~ NATURE OF PROPOSED RESEARCH:

a. ~~Staff Research~~/ **Student Research** (delete one)

b. If Student Research ..... Degree : **PhD (Software Engineering)**
Course Code : **SWEN690**

c. Project Title: **Expertise Coordination for Agile Software Development Projects**

2. INVESTIGATORS:

(a) Principal Investigator

Name :Mawarny Md. Rejab

Email address :Mawarny.Md.Rejab@ecs.vuw.ac.nz

School/Dept/Group :Engineering & Computer Science/ Software Engineering

(b) Other Researchers Name Position

None

(c) Supervisor (in the case of student research projects)

Prof James Noble

Dr Stuart Marshall

3. DURATION OF RESEARCH:

(a) Proposed starting date for data collection:

30 September 2013

(**Note:** that NO part of the research requiring ethical approval may commence prior to approval being given)

(b) Proposed date of completion of project as a whole: **30 June 2015**

4. PROPOSED SOURCE/S OF FUNDING AND OTHER ETHICAL CONSIDERATIONS

(a) Sources of funding for the project

Please indicate any ethical issues or conflicts of interest that may arise because of sources of funding e.g. restrictions on publication of results

None

(b)Is any professional code of ethics to be followed: **Y** ☐ N ■

If yes, name

(c) Is ethical approval required from any other body **Y** ☐ N ■

If yes, name and indicate when/if approval will be given

5. DETAILS OF PROJECT

Briefly Outline:

(a) The objectives of the project:

The aim of this study is to build a theory of expertise coordination in Agile software development projects, which is guided by three research questions:

- **How is expertise coordinated in Agile software development projects?**
  This question attempts to understand the expertise coordination processes that should be carried out by Agile team members in managing expertise dependencies.
- **What factors influence expertise coordination in Agile software development projects?**
  This question attempts to identify factors that should be considered in coordinating expertise in Agile teams.
- **What expertise coordination practices should be applied in Agile software development projects?**

This question attempts to determine practices that Agile team members should apply in coordinating expertise in Agile teams.

(b) Method of data collection

This study is carried out by employing Grounded Theory, which using interviews as the predominant source of data collection and also observation (Ethics Approval: 19488). Multiple sources of data however will strengthen the Grounded Theory research. Therefore, the researcher intends to include document analysis in conjunction with the interviews and observation.

The potential participants for this study are Agile practitioners who engage in Agile software development projects. Thus, this study requires the researcher to identify software companies that apply Agile practices in their software development projects. Document analysis is based on the Agile artefacts that have been produced by Agile software development teams. The document analysis will provide proofs in strengthening the other data collection methods. It also tends to provide more information about the study of the phenomena. It will assist researchers to collect more information for the next interviews.

No personal details such as participant's name or company's name will be collected during the interview session. All findings will be kept confidential and will be destroyed after the completion of the research.

(c)The benefits and scientific value of the project

- **This study will contribute theories on expertise coordination in Agile software development projects. The novelty of this study lies in the demonstration of theories related to the processes of expertise coordination, factors that influence expertise coordination, and expertise coordination practices in Agile teams.**
- This study will provide empirical evidence that will assist Agile practitioners and researchers to gain in depth understanding of the expertise coordination in Agile Software Development projects. This evidence is important to contribute to the body of knowledge in software engineering and can serve as working hypotheses that could be tested by future researchers.

- **This study will provide guidelines on how to coordinate expertise in Agile software development projects. The guidelines will also assist Agile practitioners in managing expertise resources and circulating the available expertise effectively. This guideline also will assist Agile teams to coordinate expertise and rely on roles outside the teams.**

(d) Characteristics of the participants

The participants for this study are Agile practitioners who engage in Agile software development projects. Eligibility criteria should be considered in recruiting participants for this study including Agile methods used and roles in Agile teams.

- **Agile methods**

  There is no restriction as to which Agile methods have been used. This study is open to recruit Agile practitioners even though there is no specific Agile method is used in their software projects.

- **Agile roles**

  More priority is put on Agile roles since this study requires a broad range of roles. The wide variety of roles is important to enable the triangulation of findings. Different roles will provide different insights and perspectives on expertise coordination in Agile teams.

The participants of this study should take part voluntarily.

(e) Method of recruitment

Agile Professional Network (APN) is a avenue to identify Agile practitioners from a different software organizations based in New Zealand. Attending Agile seminar and workshops which are organized by APN or software companies is another way to recruit more participants. Additional to APN, Agile conferences which run around the world can be used as a platform to recruit a wide range of participants.

The potential participants will be provided with a brief outline of the intended research via email. The researcher will contact the interested software companies to develop mutually agreeable terms for their participation in the research. The researcher will elicit their participation on a voluntary basis and all collected data will be kept confidentially.

The participants will be provided with a document that consists of the purpose and scope of this study, the time commitments expected from the participants, the processes of data collection, and other commitments expected from participants. The consent document, and information sheet are attached together to this application.

(f)Payments that are to be made/expenses to be reimbursed to participants

None

(g) Other assistance (e.g. meals, transport) that is to be given to participants

None

(η)Any special hazards and/or inconvenience (including deception) that participants will encounter

None

(i) State whether consent is for (delete where not applicable):

(i) **the collection of data**

(ii) ~~attribution of opinions or information~~

(iii) ~~release of data to others~~

(iv) **use for a conference report or a publication**

(iv) **use for a PhD thesis**

Attach a copy of any questionnaire or interview schedule to the application: **attached**

(j)How is informed consent to be obtained (see sections 4.1, 4.5(d) and 4.8(g) of the Human Ethics Policy)

(i) the research is strictly anonymous, an information sheet is supplied and informed consent is implied by voluntary participation in filling out a questionnaire for example (include a copy of the information sheet) Y ☐ N ■

(ii) the research is not anonymous but is confidential and informed consent will be obtained through a signed consent form (include a copy of the consent form and information sheet) Y ■ N ☐

(iii) the research is neither anonymous or confidential and informed consent will be obtained through a signed consent form (include a copy of the consent form and information sheet) Y ☐ N ■

(iv) informed consent will be obtained by some other method (please specify and provide details) Y ☐ N ■

With the exception of anonymous research as in (i), if it is proposed that written consent will not be obtained, please explain why

Not applicable

(k) If the research will not be conducted on a strictly anonymous basis state how issues of confidentiality of participants are to be ensured if this is intended. (See section 4..1(e) of the Human Ethics Policy). (E.g. who will listen to tapes, see questionnaires or have access to data). Please ensure that you distinguish clearly between anonymity and confidentiality. Indicate which of these are applicable.

(i) access to the research data will be restricted to the investigator Y ☐ N ■

(ii) access to the research data will be restricted to the investigator and their supervisor (student research) Y ■ N ☐

(iii) all opinions and data will be reported in aggregated form in such a way that individual persons or organisations are not identifiable Y ■ N ☐

(iv) Other (please specify)

Not Applicable

(l)Procedure for the storage of, access to and disposal of data, both during and at the conclusion of the research. (see section 4.12 of the Human Ethics Policy). Indicate which are applicable:

(i) all written material (questionnaires, interview notes, etc) will be kept in a locked file and access is restricted to the investigator Y ■ N ☐

(ii) all electronic information will be kept in a password-protected file and access will be restricted to the investigator Y ■ N ☐

(iii) all questionnaires, interview notes and similar materials will be destroyed:

(a) at the conclusion of the research Y ☐ N ■

(b)**Four (4)** years after the conclusion of the research; or Y ■ N ☐

(iv) any audio or video recordings will be returned to participants and/or electronically wiped Y ■ N ☐

(v) other procedures (please specify):

Access will be restricted to the investigator and supervisors

If data and material are not to be destroyed please indicate why and the procedures envisaged for ongoing storage and security

Not Applicable

(m) Feedback procedures (See section 7 of Appendix 1 of the Human Ethics Policy). You should indicate whether feedback will be provided to participants and in what form. If feedback will not be given, indicate the reasons why.

The participants will be continuously updated about the research and all essential findings, and provided with prompt feedback to confirm or review the researcher's interpretation of given data.

(n) Reporting and publication of results. Please indicate which of the following are appropriate. The proposed form of publications should be indicated on the information sheet and/or consent form.

(i) publication in academic or professional journals Y ■ N ☐

(ii) dissemination at academic or professional conferences Y ■ N ☐

(iii) deposit of the research paper or thesis in the University Library (student research) Y ■ N ☐

(iv) other (please specify)

None

Signature of investigators as listed on page 1 **(including supervisors) and Head of School.**

NB: All investigators and the Head of School must sign before an application is submitted for approval

.....

Date...16/9/2014/.....

Date.../15/9/2014....

.....

Date.....19 SEPT 2014.....

Head of S

.....

Date.....23/9/14.....

APPLICATIONS FOR HUMAN ETHICS APPROVAL

CHECKLIST

- Have you read the Human Ethics Policy?
- Is ethical approval required for your project?
- Have you established whether informed consent needs to be obtained for your project?
- In the case of student projects, have you consulted your supervisor about any human ethics implications of your research?
- Has your supervisor read and signed the application?
- Have you included an information sheet for participants which explains the nature and purpose of your research, the proposed use of the material collected, who will have access to it, whether the data will be kept confidential to you, how anonymity or confidentiality is to be guaranteed?
- Have you included a written consent form?
- If not, have you explained on the application form why you do not need to get written consent?
- Are you asking participants to give consent to:
  - collect data from them
  - attribute information to them
  - release that information to others
  - use the data for particular purposes
- Have you indicated clearly to participants on the information sheet or consent form how they will be able to get feedback on the research from you (e.g. they may tick a box on the consent form indicating that they would like to be sent a summary), and how the data will be stored or disposed of at the conclusion of the research?
- Have you included a copy of any questionnaire or interview checklist you propose using?
- Has your application been seen by the head of your school or department (or the person given responsibility to consider applications on behalf of the head (see section 4.5(b) of the Human Ethics Policy).

PLEASE FORWARD YOUR COMPLETED APPLICATION FORM TO THE SECRETARY, HUMAN ETHICS COMMITTEE OR, IN THE CASE OF APPLICATIONS FROM SCHOOLS OR DEPARTMENTS WITH AN APPROVED ETHICS SUB-COMMITTEE, TO THE

VICTORIA UNIVERSITY OF WELLINGTON
Te Whare Wananga o te Upoko o te Ika a Maui

Attachment A: Cover Letter for Research Participant

[Participant's name]
[Company name]
[Employment address]

Dear [Participant's name],
PhD Research : Expertise Coordination for Agile Software Development Projects

Thank you for your interest to participate in this research.

Prior to conducting the document analysis, Victoria University of Wellington requires that I obtain your written informed consent. This consent is a normal part of any research project and forms one criterion of the Victoria University Human Ethics Committee Guidelines that I must meet.

Attached herewith are:

- A copy of the research document that outlines the purpose, scope & approach to the project.
- A consent form that you need to sign and return to me [in the enclosed envelope] if you decide to participate in this research.

If you require any further information, please feel free to email me at Mawarny.Md.Rejab@ecs.vuw.ac.nz or call me on +02230034839. Alternatively you may contact either of my supervisors Prof James Noble (kjx@ecs.vuw.ac.nz or +64 4 463 6736) or Dr. George Allan (george.allan@ecs.vuw.ac.nz or +64 4 4636741)

Thank you.

Yours sincerely

Mawarny Md. Rejab
(PhD Student)
Mobile: +0223034839
E-mail:Mawarny.Md.Rejab@ecs.vuw.ac.nz

VICTORIA UNIVERSITY OF WELLINGTON
Te Whare Wananga o te Upoko o te Ika a Maui

Attachment B: Information Sheet

General Information
This research study contributes towards the requirements for a PhD degree at Victoria University of Wellington, New Zealand.

Research Topic: **Expertise Coordination for Agile Software Development Projects**

PhD Student: Mawarny Md. Rejab (Mawarny.Md.Rejab@ecs.vuw.ac.nz, 0223034839)
Supervisors: Prof. James Noble (kjx@ecs.vuw.ac.nz, +64 4 4636736)
Dr. George Allan (george.allan@ecs.vuw.ac.nz, +64 4 463 6741)

Objectives of Research
The objective of this research is to explore and understand the current issues of expertise coordination in an Agile Software Development project. This research also aims to determine factors which influence processes of expertise coordination in an Agile Software Development project. This research attempts to identify expertise coordination practices that should be carried out by Agile team members to managing their expertise dependencies. The central of this research is to explore and understand how expertise is coordinated in Agile software development projects.

Research Methodology
This research employs Grounded Theory as a qualitative research. The researcher has sought and has been granted approval by the University's Human Ethics Committee to participant observation.

The researcher aims to collect data of expertise coordination in Agile software development projects by using document analysis, with conjunction of interviews and observations. The document analysis is based on the Agile artefacts that have been produced by Agile teams involved. The document analysis will provide proofs and more information about the study of the phenomena. The Agile artefacts will be collected during interviews and observations.

Purpose of Data Collection
The collected data will be analysed to infer theories of expertise coordination for Agile software development projects. Findings from this research will be published in conference papers and referred journals for the benefit of the larger research community. The final PhD thesis will be published and can be accessed at Victoria University library of Wellington, New Zealand.

VICTORIA UNIVERSITY OF WELLINGTON
Te Whare Wananga o te Upoko o te Ika a Maui

Confidentiality and Consent

The data collected will be kept confidentially and will be destroyed at the completion of the research. All identification data collected for this research will remain confidential to the researcher, Mawarny Md. Rejab and her supervisors, Prof James Noble and Dr. George Allan.

VICTORIA UNIVERSITY OF WELLINGTON
Te Whare Wananga o te Upoko o te Ika a Maui

Attachment C: Consent for Research Participants

Research Title: Expertise Coordination for Agile Software Development Projects
Researcher : Mawarny Md. Rejab

I have been given and have understood an explanation of this research project and the confidentiality conditions. I have had an opportunity to ask questions and have them answered to my satisfaction.

I agree to participate in this study for Mawarny's PhD degree and resultant theses and publications. I consent to the collection and use of documents, diagrams, screens, pictures, and other artefacts in this research.

I understand that I may withdraw from this research up to one months without penalty or explanation by contacting the researcher or her supervisors.

Do you agree to being observed?

☐ Yes ☐ No

Would you like to receive a copy of any publications that are based on the documents analysis results?

☐ Yes ☐ No

If yes, please provide an email address, or mailing address that we can use to send a copy of the publication.

_____

_____

Name: _____

Signed: _____

Date:_____