12-9-2022

Practical requirements elicitation in modern product development: A multi-case study in discontinuous innovation

Michael King
Mississippi State University, mikeking214@gmail.com

Practical requirements elicitation in modern product development:
a multi-case study in discontinuous innovation

By

Michael King

Approved by:

Reuben F. Burch V (Major Professor)
Lesley Strawderman
Brian Smith
Kari Babski-Reeves (Graduate Coordinator)
Jason M. Keith (Dean, Bagley College of Engineering)

A Thesis
Submitted to the Faculty of
Mississippi State University
in Partial Fulfillment of the Requirements
for the Degree of Master of Engineering
in General Engineering
in the Bagley College of Engineering

Mississippi State, Mississippi
December 2022

Name: Michael King

Date of Degree: December 9, 2022

Institution: Mississippi State University

Major Field: General Engineering

Major Professor: Reuben F. Burch V

Title of Study: Practical requirements elicitation in modern product development: a multi-case study in discontinuous innovation

Pages in Study: 54

Candidate for Degree of Master of Engineering

Practical modern product development, specifically rapid, lean efforts to create new disrupting or specialized products, face constraints that require modified requirements elicitation (RE) techniques. Requirements elicitation conventions have not been updated to address the challenges of these approaches, and industry practitioners lack the tools to select the most efficient techniques. This study examines the RE approaches performed by three resource-limited teams conducting discontinuous new product development through a multi-case study to identify gaps between the literature and practice, with suggestions to fill them. Our findings suggest modern RE practices and challenges closely reflect those found by studies on RE in agile development, highlighted by a limited variety of techniques and a focus on user feedback despite user unavailability, resulting in partially complete and validated requirements. We suggest further investigation into practical technique selection, development of technique metrics, and a technique selection literature review to practitioners prior to RE.

DEDICATION

This work is dedicated to my wife Kerry, without whose enduring support and patience this thesis would not be possible. I am indebted to you for all you did to sustain for me. I am sorry I broke my promise.

ACKNOWLEDGEMENTS

I would like to thank and acknowledge my advisor Dr. Reuben Burch V, for his enthusiasm and enjoyment of education, which lead me to write this thesis, and his guidance over the duration of its completion. I acknowledge my committee, Dr. Lesley Strawderman and Dr. Brian Smith for their support and patience as I completed this study. I also acknowledge my company and colleagues whose work and words provided the invaluable data for this research.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER I

BACKGROUND

1.1 Introduction

The rapid pace of technology and market changes has increased the complexity of modern product development, which must now face volatile requirements that evolve quickly and can even become obsolete before development is over (Boehm, 2000). Requirements are especially uncertain within discontinuous innovation (i.e., discontinuous new product development (DNPD))—an area characterized by radically new, “revolutionary” or “disrupting” products—which has been further enabled by modern technology (e.g., the Internet, mobile devices, telework). The development of this type of non-evolutionary product is highly desirable at a large scale since it creates significant competitive advantage (e.g., the iPod for music listening, Netflix for watching movies and television) (Veryzer, 1998), but it is also necessary at a small scale when introducing a specialized product to existing users (e.g., a touchscreen for aircraft carrier landings). The same technology advancements that make requirements volatile have also enabled more lean and agile development, so products can be created faster with less resources. With this agility and efficiency, however, comes more iteration, less formality, and a streamlined process, resulting in new development practices that may not reflect the conventional best practices designed for evolutionary innovation. Despite the interest in discontinuous innovation, little is documented about the development process and how it is managed. And because of the volatility of requirements, how requirements engineering is performed within DNPD—specifically the requirements elicitation (RE)—is of particular interest. This paper hypothesizes that industry-based practical RE conducted by teams commonly associated with DNPD (i.e., small, lean, autonomous, and unstructured) is different than formal literature-based RE due to the unique challenges DNPD poses. To understand the RE practices engineers have adopted for DNPD, and how and why it deviates from the RE defined by literature, this research examines multiple case studies within small agile teams developing specialized non-incremental products.

1.2 Requirements Engineering

Requirements engineering is the process of capturing, organizing, communicating, and documenting the required services and constraints of a system being developed. The requirements are the capabilities or characteristics that define what the system is required to do, who its users are, what environment or context it exists in, and the constraints under which it is required to operate. The system can be anything engineered, like software, a physical product, or even an organization. The aim of requirements engineering is to determine all details that could affect system development before development begins. This goal is based on two core ideas: (1) the later mistakes are discovered in the development process, the more expensive they are to correct, and (2) it is possible to determine a complete, consistent, and relevant set of requirements before development begins. Requirements of an engineered system are always present—whether implied or explicitly provided—but incomplete or poor requirements engineering can result in a system that is unsatisfactory, expensive, or needs updating. Therefore, a good requirements engineering process should lead to a good system.

1.3 Requirements Elicitation (RE)

Requirements engineering typically involves five main activities (Kotonya 1996): elicitation, analysis, documentation, validation, and management. Requirements Elicitation (RE) is the first and most crucial phase; many problems in system development trace back to poor elicitation (Christel 1992). The term “elicitation” is chosen deliberately because requirements are not simply available to be captured but must be systematically sought out and analyzed. RE is performed by working with involved communities (e.g., sponsors/funders, developers, experts, end users, and customers) to understand the problem, identify needs, discover specifications, define system boundaries, and distill requirements to meet the communities’ constraints. To understand the problem specifically, the boundaries, stakeholders, goals, tasks, and use cases must be determined (Nuseibeh 2000). *Boundaries* specify the scope of the system and how, where, and when it fits into its operational environment. *Stakeholders* are the individuals or organizations who are affected by the system, like the customers who buy it, staff who maintain it, and users who interact with it. Often the stakeholders are highly diverse (e.g., first-time users, experts, technicians, managers etc.) and have competing interests, so determining the needs of each stakeholder class is an additional requirement dimension. And when system usability is a concern, a complete characterization of the target users is critical. *Goals* refer to the aims of both the system and its stakeholders—from high-level organizational goals down to low-level technical objectives. Since potential solutions to a problem can bias the specifications of a system, eliciting goals focuses the requirements engineer on the source of the problem and avoids solution-informed requirements. *Tasks* are the actions and behaviors of users interacting with a system. Documenting these help elicit requirements that users are unaware of or cannot articulate. *Use cases* describe the instances in which a system is used, including the motivation and uniqueness for each.

Conventional literature defines several techniques used to elicit requirements (Sood & Arora, 2012; Zhang, 2007; Zowghi & Coulin, 2005), which can be selected and combined based on the type of requirement, stakeholder, or domain, and within the time and resource constraints. Traditional techniques include data gathering methods like interviews, questionnaires, documentation analysis, domain analysis, task analysis, introspection, observation, and ethnography. Collaborative or group techniques include prototyping, joint application development (JAD), brainstorming, workshop/focus group, user scenarios, apprenticing, and simulations, to elicit requirements more interactively from stakeholders. Prototypes and modelling can be used in the absence of stakeholders or combined with other techniques to gather feedback. Cognitive techniques like laddering, card sorting, repertory grids, class responsibility collaboration (CRC), goal modelling, viewpoints, and protocol analysis are also used, especially for knowledge-based systems.

Documentation analysis involves any type of targeted fact-finding through digital or physical research and information gathering, that may discover explicitly documented requirements from existing sources. This may include the requirements extraction from published standards, datasheets, academic literature, manuals, existing products, diagrams, or media (Sharma & Pandey, 2013; Sood & Arora, 2012). An *interview* is a method of eliciting the facts and opinions held by experts, users, and stakeholders through interactive questioning or conversation (Paetsch et al., 2003). A structured interview is systematic and uses a set of pre-defined questions targeting specific answers, while an unstructured interview is open-ended and allows the interviewee to provide the feedback they feel is most important. Ultimately interviews should use a combination of pre-defined questions and open-ended discussion to provide a complete rich data set, which is their major advantage and makes them a popular method. The disadvantage is that most of the data is qualitative and often conflicting, which can make analysis challenging. Interviewing a large sample can provide a consensus, but this requires significant resources and much more analysis. A *questionnaire* is a non-interactive, structured method of eliciting facts and opinions from stakeholder in question form. A key feature is that it can more easily gather quantitative feedback by requesting a rating on a scale. Another advantage is it can be distributed widely with few resources and analyzed at scale with a consistent question set (Zowghi & Coulin, 2005). *Observation* is the act of viewing and documenting users and other stakeholders in a relevant environment within the problem domain to better understand their tasks, behavior, issues, and motivations. It provides the purest form of knowledge elicitation because it occurs with natural context and isn't obscured by a stakeholder's memory, opinion, oversimplification, or bias. Observation can occur in person or via recordings.

A *use case* specifies a sequence of interactions between the system and an external entity (e.g., a person, another system) and the conditions and environment under which the interaction occurs (Paetsch et al., 2003). Varying the conditions or environment create variants or extensions of general use cases, which can have unique interactions and lead to new requirements. By determining the details of a system's use cases, the engineers can determine how the system must function in each use case and what specifications it needs to perform well. A *scenario* is a simulated interaction session with a system in a given use case for the purpose of observing and discovering new requirements. Scenarios can use actors or be conducted on paper, but should include system start and end states, standard activities, and exceptions to the events. A *focus group* is a technique where a small group of users from varied backgrounds or skills responds to prompts and discusses a system to provide candid feedback and opinions (Paetsch et al., 2003). Focus groups are excellent at gathering user perceptions, priorities, and desires, and revealing opposing views or a consensus, which are more difficult to gather in other methods. They can also be used following prototype development to gather feedback at scale on a proposed or developing solution that leads to new or updated requirements. *Brainstorming* is an open-ended group discussion, similar to what occurs in a focus group, but the purpose is creative idea sharing to generate potential solutions, which are translated into requirements (Zowghi & Coulin, 2005). A *prototype* is an initial, often feature- or fidelity-limited, version of a system developed early in the process for evaluation and feedback. Prototypes can be used in conjunction with elicitation techniques to validate or update requirements. Especially in software and agile development environments, prototypes are created and updated continuously to evolve requirements throughout the development process. *Modelling* is a method of mapping and illustrating a problem domain so requirements can be gathered, clarified, related, and prioritized. Some examples of models start with stakeholder goals (e.g., KAOS, I*) or auto-generated scenarios (e.g., CREWS) to aid engineers in eliciting and organizing requirements (Nuseibeh & Easterbrook, 2000).

Though there is no specific RE process accepted by all developers, nor a formula to determine which elicitation activities to perform for a given problem, there are international standards developed by the International Organization of Standardization (ISO) that set some definitions for Requirements Engineering. Standard ISO/IEC/IEEE 29148:2018, “specifies the required processes implemented in the engineering activities that result in requirements for systems and software products throughout the life cycle,” “specifies the required information items produced through the implementation of the requirements processes,” and “provides details for the construct of well-formed textual requirements” (ISO/IEC/IEEE, 2018). The standard aims to be applicable for, “man-made systems, software-intensive systems, software and hardware products, and services related to those systems and products, regardless of the project scope, product(s), methodology, size or complexity," though we found no evidence in the literature that this is well-adopted. Standard ISO/IEC/IEEE 15288:2015 is useful in outlining key parts of the System Requirements definitions process, including defining system boundary conditions (e.g., organization, environment, and constraints), functions, operation modes, implementation constraints, risks, and attributes (ISO/IEC/IEEE, 2015), but it provides no guidance on how to elicit these. In fact, most requirements techniques and tools found in literature are more concerned with the specification or representation of requirements than the elicitation (Christel & Kang, 1992). The assumption seems to be that guidance is only necessary to define and document the types of information that make up the system's requirements, not on how to choose or perform the complex processes to elicit them (Carrillo De Gea et al., 2011).

There is some literature that examines the challenges and issues of RE (Christel & Kang, 1992; Davey & R. Parker, 2015; Janits, 2013; Sharma & Pandey, 2014; Sutcliffe & Sawyer, 2013). Christel and Kang's review of RE issues is the most comprehensive, though it is outdated. The problems faced during RE typically have to do with the scope, understanding, and volatility of requirements. *Scope* issues come from defining the system boundaries and context (e.g., organizational, environmental, project factors) and determining how general or specific to focus requirements: too broad or too narrow and requirements may be ambiguous, incomplete, or unusable. *Understanding* issues stem from a diversity of the perspectives, skills, motivations, articulation, and communication types among stakeholders and engineers that lead to language incompatibility, common knowledge gaps, biases, interpretation errors, and ambiguity. *Volatility* issues have to do with changing requirements due to evolving user needs, re-prioritization, and organizational complexity, inherent with any system but becoming more common with rapid technology advancements.

No one elicitation technique avoids these issues; any one approach will have varying success under different conditions and across projects. To improve outcomes and mitigate common RE issues, a deliberate context-dependent selection of elicitation techniques and an iterative process to gather, organize, and validate information is required. The process should combine multiple techniques based on how well they individually address the common issues and be revisited throughout development. According to Christel and Kang, the elicitation approach is dependent not only on the diversity and experience levels of the cross-disciplinary sources of requirements, but also the diversity of the problem being formulated, which ranges from a fully understood system to a new, novel one. They propose a methodology that tailors the use of techniques to the problem, including guidelines on how to fit the methodology to different projects (Christel & Kang, 1992).

1.4 RE Challenges in Modern Product Development

Organizations, users, and systems can change during development efforts, therefore making requirements volatile, which increases opportunities for incomplete or outdated specifications and poor outcomes. This idea is primarily discussed regarding software development efforts, but it can be shown to affect many modern development efforts. Modern development efforts have requirements that tend to evolve quickly and can even become obsolete before project completion (Boehm, 2000). Advances in technology (both the technology products and the tools to develop products quickly) and dynamic market needs have increased customer expectations and pressure to innovate and bring products to market faster (Anitha 2012). These rapid changes in competitive threats, stakeholder preferences, technology, market innovativeness, and time-to-market pressures severely challenge the development of systems based on pre-specified requirements (Boehm, 2000). And because technology has increased the pace, demand,

and accessibility of new product development, engineering teams must be smaller, leaner, and operate on shorter timelines than ever before, resulting in adjustments to development approaches, including RE activities. Agile development is one such approach created partly in response to the new challenges posed by the rapidly changing conditions of the modern world.

Agile development is an iterative approach based on continuous evaluation of requirements, plans, and results to enable rapid changes. Studies have shown that RE performed in an agile environment is different than in less iterative approaches (Paetsch et al., 2003). Agile practices, particularly flexible and reactive methods designed to stimulate customer feedback, such as lean agile approaches, are preferred for emergent and fast changing requirements (Lindvall et al., 2002). In fact, agile practices address some of the common issues with traditional RE, such as communication, overscoping, requirements validation (Inayat et al., 2015), and emphasis of domain knowledge (Daneva et al., 2013). However, agile methods do have deficiencies, including, differences in customer opinion (Daneva et al., 2013), minimal documentation, neglect of non-functional requirements (NFRs), and insufficient customer availability (Inayat et al., 2015; Pichler et al., 2006; Ramesh et al., 2007). Customer availability is especially challenging in these innovative domains where products can be developed for users, markets, or systems that don't exist yet. When there are only minimal early adopters or in the absence of actual customers, alternative strategies for requirements elicitation must be used, include finding the most proximate and relevant existing customers or mining requirements from online sources (e.g., social networks, discussion forums) (He & Yan, 2014). Startups have even adopted using engineer-generated personas and user stories to replace real users or customers (Paternoster et al., 2014), however, the information they provide to developers has been found to be insufficient (Medeiros et al., 2017).

Startups, whether they explicitly subscribe to agile methodology or not, resemble agile teams since they gather requirements throughout development, rather than in a single phase like traditional engineering-style approaches (Melegati et al., 2019). Case studies on startups have found their elicitation approaches to be primitive and informal, and occur alongside product evolution, often based on a pre-conceived idea (Rafiq et al., 2017). Upfront requirements practices are considered a waste of time and a hindrance to releasing a product as soon as possible. Formal documentation or long-term planning is avoided to prioritize development pace, reduce time to market, and validate previous assumptions regarding the customers and the product-market fit (Gralha et al., 2018). Some startups do use standard elicitation techniques like interviews, prototyping, and brainstorming, but also use less proven ones like competitor product analysis, collaborative team discussion, and use of model or surrogate users (Rafiq et al., 2017). Although startups frequently create successful products for growing markets, there is evidence that poor management of changing or overwhelming requirements can lead to startup or product failure (Crowne, 2002; Davey & R. Parker, 2015).

Sutcliffe and Sawyer examine a framework regarding the relative “knowness” of requirements that affects the level of difficulty and effort to elicit them, which is especially relevant to modern development efforts like startups and discontinuous innovation (Sutcliffe & Sawyer, 2013). The framework suggests requirements can be categorized into four groups based on whether they are *expressible* (i.e., known knowledge), *articulated* (i.e., documented knowledge), *accessible* (i.e., able to be recalled) or *relevant* (i.e., important to the problem). *Known knowns* are expressible, articulated, and relevant; *known unknowns* are not expressible or articulated, but accessible and potentially relevant; *unknown knowns* are potentially accessible but not articulated; and *unknown unknowns* are not expressible, articulated, or accessible but still potentially relevant.

In the case of an engineer eliciting requirements from a user, known knowns are the obvious ones and unknown knowns are ones the user may not realize they can provide, or have forgotten, but will if the engineer can use the right technique. Known unknowns are more challenging since they may be consciously or unconsciously withheld and require more effort, time, or creativity to elicit it, like tacit knowledge. Unknown unknowns are the hardest and most frequently missed, since they are knowledge that neither the engineer nor the user possess but ultimately impact system design. Sometimes this is knowledge from some domain outside of that of the user and engineer, but more frequently now the unknown unknowns are in new domains that simply don't exist yet. This occurs when products are developed for users, markets, or systems that are brand new, rapidly evolving, or are not available yet. Gathering this unknown knowledge from new green-field domains—compared to evolutionary brown-field domains—is the greatest challenge to requirements elicitation (Sutcliffe & Sawyer, 2013). The first step to addressing the unknowns in a new area is recognizing there may be missing knowledge and investing resources to find out if there is, and if so, where the gap lies (Sutcliffe & Sawyer, 2013). However, elicitation techniques require varied amounts of resources and, because sampling is inherently blind when knowledge is unknown, scoping the resources for knowledge elicitation is difficult. So for resource-limited (i.e., lean) development teams, choosing the most efficient techniques and sampling strategies (i.e., size, diversity) to discover unknowns is critical for effective requirements elicitation. Table 1.1 summarizes the estimated relative effectiveness of certain basic techniques.

Table 1.1 Potential of elicitation techniques for discovering unknowns (Sutcliffe & Sawyer, 2013)

Technique	*Known-unknowns*	*Unknown-unknowns*	*Articulation*
Interviews	Depends on plan	Follow up questions, sample size	Natural language ambiguity
Observation	Duration and context plan	Duration in context	Ambiguity in interpretation
Workshops	Plan and composition	Number and composition	NL ambiguity
Protocols/dialogues	Plan and analysis codes	Limited potential	Narrow, detailed analysis
Scenarios	Plan and sample	Sample size and diversity	Sample and bias, NL ambiguity
Prototypes	Design variations	Limited potential	Extent of implementation

1.5 Domain / Use Case

Conducting requirements elicitation in compliance with the rigorous conventional approaches defined in literature is often only feasible for the development of large systems within established companies. Literature-based techniques are inherently idealistic and often ignore limitations due to funding, team size, or access to customers, which affect a development team's ability to perform RE. Established companies have mature and refined processes, proven products and services, documentation practices, long-standing partner and customer relationships, market share, and dedicated resources (i.e., people, facilities, funding). Startups are new and short-lifespan organizations constantly looking for repeatable and scalable business and operate lean under significant time and market pressure, including extreme conditions of uncertainty (Ries, 2011). Like small businesses, they have limited capabilities and resources, and often specialized customers and products. Unlike established companies, small businesses and startups often do not have the ability to build multi-disciplinary teams, including RE engineers, to conduct RE during development, so the researchers or product designers must take on this additional work.

The "system development" or "product development" referred to in requirements engineering literature can mean the design and implementation of any system, from software apps to processing facilities, military aircraft to computer mice. The scope and nature of these efforts can vary drastically, yet an extensive literature review found no examples of how this affects the requirements engineering approach and the practical implications for the engineering staff. Each system, team, domain, and project has unique constraints that define their abilities and impact their success. There are countless constraints that can contribute, but some include funding type and amount; development timeline; staff size, experience level, and background; problem scope and complexity; and user base size, diversity, and level of access. For example, a computer manufacturer may have millions of dollars and multiple teams to dedicate to a new wearable device's development, but face significant scrutiny by their massive user base, whereas a specialty rehabilitation company may rely on hospital grants and four engineers to develop a new balance assessment platform, but only have a few hundred users, whom they work with regularly. Companies conducting discontinuous innovation and new product development (DNPD)—often small businesses and startups—face extreme pressure to solve challenging problems under unique conditions and are often the ones to disrupt established systems with inventive ideas. DNPD is a subset of system or product development characterized by new and inventive problems (i.e., non-incremental innovation), rapid development (i.e., short funding timelines, rush to market), user-centered design (i.e., customer involvement), a high degree of uncertainty (i.e., unknown technical and market feasibility), and nontraditional or agile-like practices (i.e., iterative design, prototyping, user involvement) (Veryzer, 1998). This type of rapid innovation is highly valued in society, evidenced by the amount of technology incubators, startup accelerators, innovation competitions, and technology grants available today.

One such funding mechanism targeted at boosting innovation in resource-limited organizations is the US Government's Small Business Innovative Research (SBIR) program. Established in 1982, the SBIR program aims to "support scientific excellence and technological innovation through the investment of Federal research funds in critical American priorities to build a strong national economy". The SBIR program is a significant source of innovation for the government and stimulus for small technology companies. The products of SBIR efforts, which are initially aimed at solving the government's most challenging problems, often end up in commercial markets (e.g., Qualcomm's first funding came from the SBIR program). Through 2019, over 179,000 awards have been given to 29,000 companies, totaling more than $54.3 billion. Awards are given in up to three phases, with the initial development phase completed in only six months with no more than $150,000. Follow on efforts typically have about two years and up to $1M. These projects span every technology domain, but all have the common goal of rapidly developing a specialized system or product with limited funding and time, making them structurally and technically similar to startups and agile teams. Due to their similarities, results from case studies characterizing RE challenges in small DNPD efforts funded by the SBIR program should have strong relevancy and broad applicability to these small rapid innovation efforts.

CHAPTER II

METHODS

2.1 Case Study Background

This research takes a multi-case study approach. Case study research focuses on understanding the dynamics present within a setting and attempts to examine a phenomenon in its real-world context (Eisenhardt, 1989). Using real-life case studies helps to improve understanding of RE, because despite what theory and literature shows to be the best approach, that approach cannot always be taken due to the challenges imposed on development by real-world constraints. Requirements elicitation is also a context-dependent, social-political process involving human interactions, so no two processes are the same (Ramesh et al., 2007), and the specific contextual conditions are pertinent to its characterization (Yin & Davis, 2007). To understand the challenges with real-world requirements elicitation and the anticipated deviations from literature-based best practices, RE practices must be examined in their natural settings. Therefore, the case study research strategy is most appropriate for the study of practical requirements elicitation in discontinuous innovation and new product development. The requirements elicitation practices within three SBIR-funded innovative research and new product development projects were examined. The cases are well-defined by their project scope and customer access challenges, while highly generalizable by their innovation challenges and resource limitations, encouraging broader applications of their findings to lean new technology development efforts.

2.2 Research Questions

This study will lead to the understanding and characterization of the practical RE approaches performed by resource-limited teams conducting DNPD. Case studies and industry experience reports highlighting the contribution of RE on projects, especially those focused on new fields, are of interest to be studied, and more collaboration between researchers and practitioners is required to both evaluate current approaches and develop new ones to address emerging problems (Zowghi & Coulin, 2005). Only prior work by Harris examines the gap between literature and practice, though their focus was on whether RE is truly as effective in practice as theory presumes (Harris, 2006). Few studies have examined RE approaches within small businesses, startups, and DNPD efforts (Gralha et al., 2018; Melegati et al., 2019; Rafiq et al., 2017; Ramesh et al., 2007; Tripathi et al., 2018; Veryzer, 1998), and none have focused on the challenges present within SBIR-funded development efforts due to their unique constraints. The researchers of this study hypothesize that the combination of rapid and lean development with the uncertainty of discontinuous innovation results in requirements elicitation techniques that deviate from literature-based approaches. Through an analysis of multiple examples of practical RE, this study aims to (1) understand and define the approaches used in practice, (2) identify how (both qualitatively and quantitatively) the practical approaches differ from literature-prescribed best practices (i.e., where a gap exists between convention and practice), and (3) seek future opportunities to inform the literature or inspire better guidance on elicitation techniques under the constraints of modern innovation. The explanatory analysis suggests theories for why the gaps exist, how they may be filled, and how these lessons can be used to inform future approaches. The study does not aim to measure or compare the effectiveness of methods, which would suggest there is a defined means to measure the success of RE activities.

The main research questions that guide the empirical study are: (1) How and why do the practical RE activities performed in industry for DNPD deviate from literature-prescribed RE best practices? And (2) are there opportunities to shrink this gap (i.e., adjust our understanding of best practices or improve practical approaches)? To focus the research on topics that may reveal answers to the research questions, the study follows five theoretical propositions, that question whether: (1) development speed affects the depth and breadth of RE; a shorter timeline means some requirements will be missed, (2) lean development requires efficient use of resources, so fewer engineers perform RE and less RE is performed, (3) systems for specialized use have fewer and less accessible users, (4) eliciting requirements to develop products for brand new systems requires creativity, and (5) engineers in product development have varying levels of knowledge in performing RE.

2.3 Study Design

A case study is an empirical inquiry that examines a contemporary phenomenon in depth and within its real-world context. Case studies are the preferred research method when “how” or “why” questions are being posed, the investigator has little control over events, and context is critical to understanding the phenomenon being studied (Yin, 2014). Evidence from multiple cases is often considered more compelling, and a multi-case study is therefore regarded as being more robust (Herriott & Firestone, 1983). The design of this study follows Yin’s direction in *Case Study Research: Design and Methods* (Yin, 2014), which is the most procedural and rigorous approach of the three prominent methodologists (Yazan, 2015), and has been updated regularly. Stake (Stake, 1995) and Merriam (Merriam, 1998) outline more naturalistic approaches that prioritize qualitative-only philosophical and contextual descriptions, which have not been updated in over twenty years to address modern development methods and are not well-suited for this study.

This research is a collective case study (compared to intrinsic or instrumental) (Stake, 1995), aimed at achieving representation through a set of similar cases that characterize a problem and make generalizations from it. The cases studied take place in a naturalistic setting (i.e., not controlled) and makes observations on participants to create a holistic understanding of the setting and problem domain. The real-life context and abundant phenomena of case studies makes them technically distinct from other experimentation, with more variables of interest than data points. To produce valuable findings with so many variables, multiple sources of evidence are used so data converges. The choice to examine multiple cases follows replication logic; each case can be selected to either (1) predict similar results (literal replication) or (2) produce contrasting results for predictable reasons (theoretical replication) (Lee & Rine, 2004). The questions posed by this study are answered through the examination of three cases with replicated conditions. Replication logic is preferred here over sampling logic, which is commonly found in survey research and used to determine the frequency of a phenomenon. Sampling logic requires a statistically significant sample so that results represent an entire population, with statistics used to establish confidence intervals of accuracy. With small diverse populations such as those studied here, statistical significance is not possible and phenomenon frequency is not the concern.

Per Yin, a case study should be designed to have five components: (1) research questions, (2) propositions, (3) a unit of analysis, (4) logic linking the data to the propositions, and (5) criteria to interpret the findings (Yin, 2014). 0 summarizes the components of this study's design.

Table 2.1 Components of the Study Design per Yin's Case Study Research: Design and Methods

Component	**Definition**	**Element**
Research Questions	Guide the study	1. How and why do the practical RE activities performed in industry deviate from conventional RE best practices? 2. Are there opportunities to shrink this gap (i.e., adjust our understanding of best practices or improve practical approaches)?
Propositions	Areas of interest to be examined that focus the research on topics that may reveal answers to the research question	• Development speed affects the depth and breadth of RE; a shorter timeline means some requirements will be missed • Lean development requires efficient use of resources, so fewer engineers perform RE and less RE is performed • Systems for specialized use have fewer and less accessible users • Eliciting requirements to develop products for brand new systems requires creativity • Engineers in product development have varying levels of knowledge in performing RE
Unit of Analysis	Scope or boundary that defines the case parameters and its specificity, so only the relevant information is collected and examined	• Requirements elicitation activities of a Phase I development effort under SBIR funding performed by a development team at a single small technology company • SBIR efforts are explicitly bound by a timeline, funding amount, personnel, solicitation, and deliverables, which define the scope of each case
Logic Linking Data to Propositions	Analytic techniques chosen to perform the analysis	• Examine the data of each case, find patterns, compare findings across cases, and test theoretical propositions o Pattern matching o Explanation building o Cross-case synthesis
Criteria to Interpret Findings	Measures used to determine the significance of the results	• Explore alternative explanations (focuses data collection, increases validity): o Requirements are missed because the engineers are incompetent o Only a few engineers at the company know how to perform RE o Engineers didn't attempt to find users o The wrong RE methods were used

2.4 Data Collection Methods

Data collection methods should be based on the research setting and question type. In this study, the guiding research question and setting requires a complete picture of the RE activities performed and the results of the RE efforts, which are defined as the subjective quality and completeness of the requirements. Since these efforts occurred in the past and span multiple participants, locations, and months, data collection cannot be done through direct observation. However, due to the nature of modern RE activities requiring a variety of methods and mediums to gather and analyze requirements, accessing the participants' digital archives and conducting interviews with them should be sufficient, if not more effective. Feedback received from participants regarding the quality and completeness of the RE process will augment and validate the collected data. And critically, the researcher directly participated in all three cases, so they bring firsthand knowledge of the RE activities and are best suited to discover the most valuable documentation.

Case study evidence primarily comes from six sources: documents, archival records, interviews, direct observation, participant-observation, and physical artifacts (Yin, 2014). Each data source has its strengths and weaknesses, but they are often complementary so using as many sources as possible produces the best results. Based on the availability of potential sources for this study, data collection shall use documents, interviews (both in-depth interviews and surveys), physical artifacts, and participant observation. Direct observation, which includes passive firsthand observation during project execution, is commonly relied on within case study research. However, since these cases examine RE activities conducted a few years prior to the creation of this study, primary data collection is limited to the gathering and organizing of documentation evidence and interviews with those who participated in the projects. A researcher who had a role in these RE activities can use their knowledge of the RE activities to guide data collection, substantiate participant responses, and provide an autoethnographic context to the analysis. Other participant observations can be gathered in the form of documentation (i.e., the notes of engineers who participated in these projects, whether directly conducting RE or observing the activities and interacting with the information) and interviews (i.e., conducted with the engineers during the study). This study will collect both quantitative and qualitative data about the processes and outcomes within each case. Multiple data collection methods and types enables synergy and triangulation of the data, which can provide stronger substantiation of constructs and hypotheses. The combination of qualitative with quantitative evidence creates a multidimensional dataset that produces a better understanding of the case and leads to a richer result. Quantitative data can expose relationships which may be missed with only qualitative evidence, and it can refute or corroborate surprising findings in qualitative data. Qualitative data can provide context or rationale to explain relationships revealed in quantitative data or directly suggest theory which can be supported by quantitative data (Yin, 2014).

The participating engineers shall be interviewed to gather information regarding their background, project role, RE process details, and RE challenges for the project(s) they worked on. The participants will also be asked to provide their opinions regarding the quality and completeness of the RE process with a 5-point Likert scale.

Data collection will follow Yin's three guiding principles to ensure construct validity and reliability: use (1) multiple sources of evidence aimed at convergence on the same facts, (2) a case study database (a formal assembly of evidence distinct from the final case study report), and (3) a chain of evidence (explicit links among the questions asked, the data collected, and the conclusions drawn).

For explanatory case studies, data collection requires gathering, recording, and presenting evidence. To begin the data *gathering*, the researcher will use their knowledge of each case to identify a list of participants and data sources to target for discovery. Data gathering will involve identifying, locating, documenting, and organizing all archives, case inputs, and materials evidence from each case. The evidence can come from many sources, including participant hard drives, cloud-based repositories (e.g., Microsoft OneDrive, Outlook, Teams), paper notebooks and folders, and materials bins, and exist in many formats including spreadsheets, documents, emails, hand-written notes, sketches, images, videos, CAD models, physical prototypes, and off-the-shelf components. Following initial data gathering, when case context can support effective questioning, questionnaires will be developed to gather qualitative data from stakeholders. To add metrics to descriptive accounts, we will ask stakeholders to rank, order, or assign scores to assess quality or completeness of various RE activities via interview or questionnaire. Questions can be posed via email, in person, or via virtual interview. This quantitative data will add a dimension that enables the application of replication logic (i.e., confirming or refuting observations) to strengthen the construct validity of the findings. To *record*, all digital archival data gathered from participants will be categorized and stored in a digital catalog (i.e., database) along with valuable details or categories relating to each item. Physical artifacts and abstract items will be documented through photo and/or textual descriptions. Questionnaire responses and recorded interviews will be cataloged with the digital collection. For all data items, the content, context, date, and quantity will be recorded in a chain of evidence spreadsheet where it can be easily organized and analyzed. To *present* the data, the digital catalog will be organized into folders corresponding to appropriate levels of detail (e.g., project, task, participant, evidence type). Spreadsheets of data will be organized into sheets and have columns that specify critical data parameters (e.g., event name, date performed, participant name, outcome, notes) that enable filtering, sorting, construct validation, and analysis. All large or abstract data items, (e.g., images, interviews, documents, prototypes) will be summarized with text descriptions for faster recall.

Since this case study examines engineers conducting their own research and RE activities, which involves its own data collection, we will make a distinction between *case data* collected about the case and its participants for this study and the *RE data* participants used to conduct RE activities during their projects. The categories of data this study shall gather are defined as follows:

Table 2.2 Data collection plan; evidence to collect

Data Type	Category	Examples

Archives and Documents	Case	• Participant communication
		  ○ Emails
		  ○ Messages (e.g., Teams, text)
		  ○ Meetings and calls
		• Participant notes
		  ○ Design notebooks
		  ○ Meeting notes
		• Participant plans
		  ○ Calendars
		  ○ Meetings
		  ○ Travel

Observations and Case Inputs	Case	• Company domain background and experience • Participant background and experience level • Problem description, project objectives, topic background, stated stakeholder concerns • Stated functional and design requirements • Prior work products or data for reuse

Interviews and Questionnaires	Case	• Questions formulated to gather data from stakeholders regarding their actions and experience of the process
		  ○ Responses provided written or recorded via interview.
		  ○ Stakeholders may include engineers, engineers, managers, SMEs, or customers

Materials	RE	• Emails (e.g., with customers, experts)
		• Meeting and call notes
		• Tables and matrices
		  ○ Functional requirements list
		  ○ Design requirements list
		  ○ House of quality
		• SME interviews
		  ○ Questions and responses
		  ○ Images
		  ○ Videos
		• Sketches
		• CAD models
		• Prototypes and products
		• Feedback/analyses/test results of prototypes/products

Artifacts	Case/RE	• Images and videos of prototypes • Prototypes • Products

2.5 Analysis Approach

Data analysis involves examining, organizing, testing, or otherwise recombining evidence, to understand phenomena and draw empirically based conclusions. Analysis is the least developed and most difficult aspect of case studies, with no formula or equation guiding the approach (Yin, 2014). However, Yin suggests four analytic strategies that help define priorities for what to analyze and why: (1) rely on theoretical propositions, (2) develop case descriptions, (3) use both quantitative and qualitative data, and (4) examine rival explanations. Relying on theoretical propositions and combining quantitative and qualitative data are the two strategies that fit these cases best. The theoretical propositions this research relies on are reflected in the research questions, specifically, that context constraints can limit the scope of RE activities for discontinuous innovation and new product development and therefore RE within some modern new product development efforts has evolved to be a departure from conventional approaches. This approach, to follow the ideas that lead to the case study and define its objectives, is a preferred analysis strategy. It helps shape the data collection plan by prioritizing and focusing attention to the relevant data that could support the proposition or alternative explanations to be examined. For example, the analysis will look for places where researchers struggled, failed, or succeeded and their causes, including features of the case that may lead to those outcomes. The second strategy, combining quantitative data and qualitative data, is employed to explain or test the theoretical propositions by adding metrics to the descriptions of the observed phenomena. Specifically, metrics will be collected for quantities of things such as funding dollars, users, engineers, timeline months, requirements gathered, interviews, and requirements updates, and participant-based subjective scores such as user input level, user access level, requirements quality, completion level, and output quality. Theoretical propositions enable analytic generalization, which, in contrast to statistical generalization that uses formulas to determine confidence based on sample and variation, uses within-case and cross-case replication to validate and generalize findings to broader theories. Each case can be considered its own experiment, with the findings supporting its own generalizable set of theories which can be compared and validated through literal or theoretical replication within the multi-case study.

Of Yin's five specific techniques proven for analyzing case studies, this study will use pattern matching, explanation building, and cross-case synthesis to examine the data of each case, find patterns, compare findings across cases, and test theoretical propositions. Pattern matching compares an empirically-based pattern with predicted ones—either the expected pattern or one of the alternatives. Explanation building specifies presumed causal links to explain a phenomenon, or how or why it occurred. Cross-case synthesis is a multi-case approach that aggregates and compares findings across individual cases, often in tables for comparing many cases. Within-case analysis will occur first. Each case will be summarized, and all variables, metrics, and key observations will be compiled so patterns and key findings can be identified within individual cases. Then, for cross-case synthesis, data from each case will be organized into dimensions or categories so it can be compared across cases. By comparing patterns measured and observed across multiple cases, this study reduces effects of biases that may exist in single cases, goes beyond first impressions, and maximizes reliability, accuracy, and internal validity.

2.6 Case Descriptions

This study examines the practical requirements elicitation activities within three SBIR-funded innovative research and new product development projects performed by a US-based small technology company between 2017 and 2021. These projects had similar characteristics, including a fixed budget and timeline, new specialized technology to improve situational awareness, a military operational environment, only a few engineering staff, and uniquely challenging user bases. The three cases are uniquely defined by the same SBIR-based scope and funding, which is a formal contract with the government that includes the timeframe, budget, personnel, inputs, and deliverables. Projects were selected with similar characteristics to limit the variability between cases to only effects of the development process, rather than differences in the project's scope or nature, and produce valuable results with fewer data points. They were also chosen because they share characteristics with modern lean product development efforts, such as short timelines to reach a minimum viable product; small team size with varied experience level and background; new or emerging problems; and issues with user base access. Though only a few engineers worked on each case, and a few worked on more than one case, this is not uncommon for small R&D companies that practice lean development. None of the cases examined were selected for further funding, which suggests the RE challenges faced by projects of this type may affect project success; however, measuring RE effectiveness using project success metrics is not within the scope of this study. The outcomes of this research provide examples of phenomena seen across the domain, which are valuable for discovering common patterns or concerns, and suggestions for improving both the literature and practices to update the understanding of RE and produce higher quality and more complete requirements. Findings should be generalizable to future product development efforts defined by the technical challenges of discontinuous innovation and resource limitations. The applicability of these cases among similarly constrained product development efforts, and the unique access granted to them for research, makes them ideal candidates for a multi-case study analysis.

Projects are coded as Project S, T, and W. Project S was to develop a new product for Army rotary wing aircrew that added capability as part of a system being developed but not yet fielded.

The system would be for all aircrew of which there are multiple roles, based on various mission profiles (e.g., transport, CSAR, MEDEVAC, combat etc.) and aircraft (e.g., UH-60, CH-47, and AH-64), though the total potential user base is less than 5000 aviators. Project T was to develop a product used by landing signal officers (LSOs) that modernized an existing system. LSOs are a subset of fixed-wing Navy pilots that receive additional training to guide fellow pilots to the deck of aircraft carriers using a specialized communication, display, and landing system. This system only exists on American aircraft carriers (and a few training systems), of which there are only eleven in the world. Project W was to develop a system that would offer a new, not yet fielded, modality of situational awareness alerts for Air Force rotary wing pilots, with potential application for all pilots. The system had been under development, so some users were familiar with it, but it needed to be improved to meet their needs and validated for broader use.

Table 2.3 Summary of the three SBIR-funded discontinuous innovation projects performed by a single small business between 2017-2021.

	Project S	Project T	Project W
Customer	Army	Navy	Air Force
Primary User	Rotary wing aircrew	LSO pilots	Rotary wing pilots
User base	< 5,000	< 350	<10,000
Engineers	3	2	3
Timeline	6 months	6 months	6 months
Funding	$112,000	$125,000	$250,000
SMEs/inputs	3	4	3
Deliverable	Physical prototype	Physical prototype	Physical prototype
Dates	Sept 2020 - Mar 2021	May 2017 - Oct 2017	May 2020 - Dec 2020
Funded Further	No	No	No
# Requirements	89	31	75
RE Process Quality	4/5	4/5	3.3/5
Req Completeness	4/5	3/5	2.67/5
Challenge	3/5	3/5	3.3/5
Challenge Uniqueness	2/5	2/5	3/5

2.7 Case Study Execution

Data collection began by gathering background information on the three cases, including the customer-provided problem descriptions, the developer-created proposed solution and plans, the participating engineers who conducted the requirements elicitation and development, and the location of data sources. Artifacts, media, documents, and communications were gathered from physical and digital sources, reviewed for relevance and content, described or summarized if necessary, organized into folders based on type and project, and catalogued in a case study database. This project data provided the context needed to generate questions for the participating engineers, whose responses could corroborate and elaborate on the cataloged evidence to provide a chain of evidence and construct validity. IRB exemption was obtained prior to gathering information from participating engineers (Mississippi State University IRB Protocol ID # IRB-22-147). Five engineers were interviewed in a semi-structured format and given questionnaires to provide opinion-based ratings. Responses were summarized, cataloged, organized, and analyzed with the other data to enable within-case and cross-case analysis. Convergent data (i.e., documentation and multiple responses sharing the same results) were identified and highlighted, while non-convergent data were noted and summarized but not included as a valuable result. Some responses included causal links that provided explanations or opinions on why certain phenomena occurred, while other links had to be inferred from the case materials. Quantitative data from Likert scores in the questionnaire—though small in sample size—provided valuable within-case and cross-case comparisons to support or negate qualitative responses. Patterns emerged where data converged within and across cases, and, when combined with the context of the responses and observations, highlighted a few key themes which make up the basis of these findings.

CHAPTER III

ANALYSIS & INTERPRETATION

3.1 Analysis

Table 3.1 Comparison of RE approaches from conventional literature (Kotonya & Sommerville, 1996, 1998; Nuseibeh & Easterbrook, 2000), findings from startups and agile methods (Gralha et al., 2018; Melegati et al., 2019; Rafiq et al., 2017; Ramesh et al., 2007; Tripathi et al., 2018), and the case studies examined here.

	Conventional	**Startups & Agile Methods**	**Case Studies**
Practitioners / Training	Team of systems engineers / Trained	Shared between founders, software engineers, marketing, sales / Untrained	2-3 engineers or scientists / Half are trained
Formality	Formal	Informal	Informal
Documentation	Recordings of all techniques, standardized requirements document	None	Interview recordings, requirements table
Techniques	interviews, questionnaires, documentation analysis, domain analysis, task analysis, introspection, observation, ethnography, prototyping, joint application development (JAD), brainstorming, workshop/focus group, user scenarios, apprenticing, simulations, laddering, card sorting, repertory grids, class responsibility collaboration (CRC), goal modelling, viewpoints, and protocol analysis	interviews, prototyping, and brainstorming, competitor product analysis, collaborative team discussion, models, surrogate users, ad hoc techniques	literature reviews, informal domain analysis, prototyping, interviews, simulation, informal task analysis, in-person observation, heuristic review, questionnaire, creative ad hoc techniques
Timing / Timeframe	Upfront, prior to development / Months to years	Iterative, if at all / Days to months	Continuous / Months
Stakeholder Involvement	Sample of population, volunteer or incentivized	Some or none, volunteer	Fewer than 5, volunteer
Validation / Completeness	Achieve consensus or consistency	Iterative/continuous through testing	Limited/none

The RE performed in these case studies was informal and limited to a few common techniques, but it was given significant priority, consumed a large percent of the development efforts, and the results were fair given the time, personnel, and funding constraints for these projects. The small group of engineers performing RE on each project had varied backgrounds, with only half having any formal education or training. Those formally trained had degrees in human factors engineering, prior roles in human factors, and experience performing RE in the past for 30-40 product development efforts. The other engineers had backgrounds in science or engineering fields within informal training in only interviewing and literature review techniques while working for the company responsible for executing the SBIR projects assessed in this research study. However, even those formally trained in RE did not perform formal RE, due to the time and funding constraints for these types of projects, which only have enough for 100-200 total labor hours per month. For example, a scientist with a Human Factors background and 17 years of experience doing RE in PD claims, "for every product I've developed outside of an academic setting—all early-stage R&D, most with limited funding—you don't have the time or the funds to follow the formal methods. Formal methods are time-consuming, require lots of access to subject matter experts, and are hard to show [customers] value immediately." But these less formal approaches can be effective at eliciting most user and technical requirements (Paternoster et al., 2014).

Regardless of their experience, all engineers in these cases believe RE is highly important (4.8/5) in the product development process. All three RE efforts used literature reviews (e.g., documentation analysis of publications, domain doctrine, standards), informal domain analyses, semi-structured interviews, prototypes, and creative ad hoc techniques. None of these techniques were documented per doctrine or formal guidelines, though requirements were always cataloged in a table that was updated continuously throughout development. Techniques uniquely used in only one project include simulation, informal task analysis, in-person observation, heuristic review, or a questionnaire.

Common to all efforts was the claim that elicitation occurred iteratively and continuously up until the project end, and requirements gathering accounted for about one third of the total project effort. However, all engineers believed the set of requirements created was incomplete (3.0/5) and blamed running out time as the cause for not being able to gather a complete requirements list. For example, one participant stated, "We wanted to interview many more [users], at least 5, and would have wanted to do more than one interview per person, but could not because of access and timing." Stakeholder access and responsiveness were significant issues for the two projects that had such specialized user populations that the customer had to supply them from their network, evidenced by the email communication and documents, and validated in the interviews. The project which used paid consultants to access users had no issues gathering requirements from them and their network, though they note that they were not able to gather requirements from the original equipment manufacturer (OEM) of the integrated system, which limited project success. Engineers cited incentive for users as having a strong correlation with their participation, "[we had to] rely on their desire to help or a suggestion by superior, which isn't always there, especially if they don't know about future systems." Those being paid, required by a superior, or already dedicated to process improvements were most accessible and responsive, compared to volunteers. For example, one project contacted over twenty users, had six respondents, and only the three customer-supplied users with ties to the development of tangential systems agreed to interviews. This resistance to participation and potential change is a recurring issue in RE (Janits, 2013; Pichler et al., 2006). In these cases, engineers believe participation was limited by barriers related to a military population (i.e., a protected population under their own Institutional Review Board (IRB) and less industry ties), a need for special knowledge (i.e., small subset of total population), and a proxy handling user communication (i.e., the customer passing on contacts in their network).

For the users or experts interviewed within each case, responses always included conflicting preferences, which were resolved by comparing against competing requirements and prioritizing either the preference of the customer or the most experienced expert. Though with so few experts interviewed for each project—only four per project—requirement accuracy and priority could not be fully validated. In fact, for all three cases, incomplete or incorrectly prioritized requirements are cited as a key reason they believe projects did not receive further funding. The engineers blame incomplete requirements on the short, fixed, arbitrary project deadlines that limit time to elicit requirements. Accessing users (e.g., identifying, contacting, receiving responses, performing elicitation, and following up) takes time and must begin before development can start, which results in shortened or rushed development. Researchers found that when the target deliverable is a minimum viable product or proof of concept, the short timeline encourages a focus on only the types and sources of requirements that enable prototype development and the appearance of progress, while deprioritizing those related to commercialization, manufacturing, and organizational or financial goals; "with more time, we could have included certain requirements we knew might exist but had to be ignored until later in development when we'd have more time to address them." A short timeline also limits the amount and diversity of users and experts that can be used to elicit requirements, necessitates informal and ad hoc techniques which can be performed faster, and leaves little time to review, refine, validate, and prioritize the requirements. Though the rigid six-month timeline in these cases is a unique aspect of the SBIR program, the engineers mostly agree that the other challenges they faced during elicitation regarding user access, small populations of specialized users, conflicting requirements, user biases, and discontinuous technology were not unique, but common to all modern product development.

Ultimately, engineers rate their RE processes as being performed fairly well (3.6/5), but because requirements were never complete (3.0/5 complete), they believe there was room for improvement in their process, more requirements to gather, or further refinement and prioritization needed that would have increased the chance of product success.

3.1.2 Discussion

Theoretical propositions guided this study to probe for where gaps may exist between practice and convention, and through pattern matching, explanation building, and cross-case synthesis, suggested rationales are provided for each identified gap, which are detailed below.

Table 3.2 Summary of gaps and their suggested rationale

	Gap	Suggested Rationale
Deviation from Convention	Limited technique use in practice	• Only easy, familiar techniques used • Technique selection frameworks unknown • No consideration for technique efficiency or effectiveness to elicit unknown knowledge
	In practice, requirements are not complete	• Short timeline; ran out of time to complete • By focusing on user requirements, strategic requirements were missed
	In practice, requirements not properly validated	• Short timeline; no time for follow up • Small sample of users; hard to cross-verify
	Documentation does not meet conventional standards	• Small team, short timeline; verbal communication often sufficient • Engineers not all formally trained • Informal and ad hoc techniques used
Unaddressed in Literature	Conventional techniques assume no user access issues	• User access issues associated with short timeline, unincentivized users, and specialized domains with small user populations • Many techniques rely on unlimited user access
	Literature ignores resource constraints	• Many RE techniques are time- and personnel-intensive, which are major constraints in industry • Resource constraints limit completeness of RE

Technique Use. Findings indicate that RE techniques used under the constraints of these case studies are limited to those that (1) only require a few stakeholders (because only a few are available), (2) can be done by only one or two engineers, (3) are fast to perform or get results, (4) are free or inexpensive, and (5) are already known by the engineer. Literature reviews, informal domain analysis, and interviews are techniques that are already familiar to many engineers from a range of backgrounds and require no formal training to quickly produce good results (i.e., many requirements) with individual effort and little to no cost (if interviews are virtual with volunteer stakeholders). Using primarily the same small subset of possible techniques is not necessarily counter to the literature—there are no prescribed technique quantities or variations—but it does seem to ignore the literature that lists techniques and their pros and cons (Sood & Arora, 2012;

Zhang, 2007; Zowghi & Coulin, 2005). It also suggests that because engineers are unaware and untrained on more techniques, they do not know if they are using the most effective technique to elicit the type of information they are looking for. For example, using Sutcliffe's framework comparing techniques' potential to elicit requirements in green-field domains (Sutcliffe & Sawyer, 2013), or Wellsandt's qualitative metrics for common techniques (Wellsandt et al., 2014), would have introduced new approaches which may have elicited more complete requirements.

Requirements Completeness. Responses from engineers regarding the completeness of the requirements indicate there were missing requirements from key stakeholders that affect product success regarding manufacturing, business and strategy, adoption, and commercial market fit. It seems the engineers focused on gathering the technical and user-centered requirements that enable prototype development and promote user approval, but by neglecting to elicit requirements from other stakeholders (e.g., system integrators, organization management, commercial customers), the product developed from incomplete requirements was not positioned for success. In following agile-like approaches to allow user feedback to guide development, it is logical that the direction can lead somewhere not supported by business goals, market needs, or manufacturability. This need to define and elicit requirements from all stakeholders related to the product is well documented (Christel & Kang, 1992), and methods for gathering "corporate" requirements outside of end-user requirements have been developed (Gershenson & Stauffer, 1999), but they remain a challenge with agile methods (Pichler et al., 2006).

Requirements Validation. Contrary to the emphasis put on requirements validation and prioritization in conventional literature, and the evidence that agile methods improve this process through continuous customer interaction (Inayat et al., 2015; Racheva et al., 2010), we found that there was little validation or prioritization conducted on the requirements. Engineers cited this as a limitation due to the time constraints and small pool of users, and that more time would have allowed more follow up for prioritization and access to more users for validation through consensus.

Documentation. Though we found that engineers highly value requirements in the product development process, contrary to empirical findings on startups (Gralha et al., 2018; Rafiq et al., 2017), there is a departure from conventional doctrine in the lack of process formality and documentation, which aligns with findings on RE in rapid PD (Gralha et al., 2018; Inayat et al., 2015). Conventional RE prescribes formal requirements reporting with specific sections and formatting, sometimes per an industry standard (Kotonya & Sommerville, 1998). While all cases used a table to record and organize requirements that was updated continuously, the tables had varied levels of detail and organization, and no other documentation was found to gather or organize the data (e.g., customer needs, project goals, needs hierarchy, house of quality, etc.). Reference publications, domain images, prototypes, and detailed process descriptions were included in status and final report documents, but the focus of this documentation was the progress made on the proof-of-concept, not on the requirements and their refinement, validation, or prioritization. Like studies on agile methods, we found that verbal communication and constant feedback reduce the documentation burden in RE, which is beneficial for small teams and short timelines. However, minimal documentation is also cited as a primary challenge to RE (Ramesh et al., 2007), especially with sudden changes to requirements, user unavailability, and significant project complexity and scope (Inayat et al., 2015).

User Unavailability. Though conventional literature does not address concerns about user availability and responsiveness, or the risk they pose to adequately developed requirements, studies on agile requirements elicitation highlight these issues (Inayat et al., 2015; Pichler et al., 2006; Ramesh et al., 2007), and this study supports those findings. The problem is, agile methods, which are most suitable for shorter development timelines and responding to dynamic markets like the ones studied here, are based on frequent interaction with potential, ideally co-located, users. This dilemma raises the question if agile-like methods for RE are an acceptable choice when user access is limited. If not, what approaches are best?

Resource Constraints. Time was cited by engineers as a primary constraint that limited the completeness and quality of requirements in these cases. The findings show that the established deadline was sufficient to produce a proof-of-concept of a product, but because there was not sufficient time to elicit an approximately complete set of requirements, that proof-of-concept was developed with incomplete requirements and ultimately was not selected by the customer for further development. When the literature discusses issues of scope in the requirements elicitation process, the concern is bounding the system, not the process (Christel & Kang, 1992), and we found little mention in the literature indicating that time to perform elicitation should be a significant factor in technique choice. The best example of RE technique guidance is Wellsandt's evaluation criteria that measures techniques regarding their application and quality for eliciting user feedback from in-use products (i.e., not DNPD) (Wellsandt et al., 2014). Their effort per user (EPU) criterion considers the technique's application cost in time (as well as personnel and materials) on a low/medium/high scale, and in the number of users needed on a few/many scale, which they've used to measure eight common techniques. Knowledge and use of this technique evaluation, including the other metrics of quality, would be valuable for engineers to use when selecting techniques within resource-limited PD efforts.

3.1.3 Suggestions

The requirements elicitation practices observed in this study are based on conventional approaches, though they share some characteristics with agile methods, and many of the challenges recorded here have been documented in literature. However, as expected, these modern product development cases are limited by real world constraints and face unique challenges not fully addressed in the doctrine, resulting in some practices that stray from conventional approaches. Here we make suggestions to enhance the literature to reflect industry practice and increase the utility of the literature for practitioners.

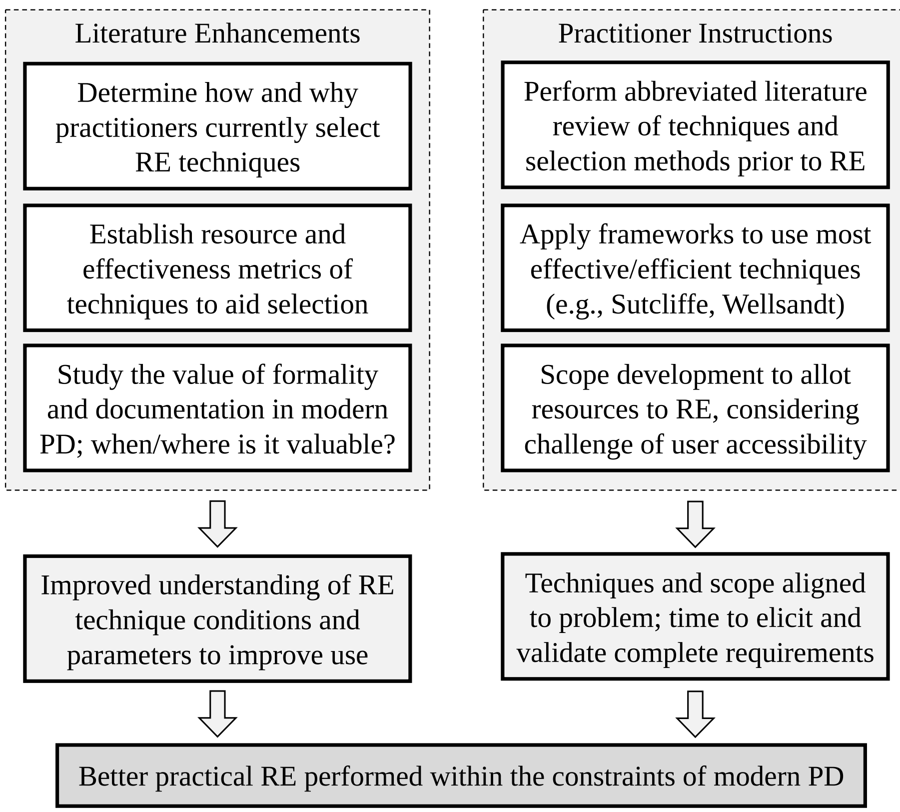

Figure 3.1 The findings of this study suggest a two-part approach to align literature and practitioners by both researching the conditions and parameters of RE techniques, to prescribe and optimize their use, and instructing practitioners to become better prepared for RE activities.

To improve elicitation effectiveness within the resource constraints and increase the variety of elicitation techniques used, RE practitioners must be better informed about techniques. Many publications list, describe, and compare techniques (Pacheco et al., 2018; Sood & Arora, 2012; Wellsandt et al., 2014; Zhang, 2007), but our findings indicate that unless engineers are formally educated in RE during their studies, RE knowledge is learned on the job, where these publications are not considered. Though it is possible a limited variety of techniques were deliberately used because they were optimal, missing exposure to all available techniques is more likely the reason. There are studies of the technique selection factors considered in practice by experts (Hickey & Davis, 2003) and in certain industries (Alzahrani, 2020; Anwar & Razali, 2012), but further work in similarly constrained new product development is needed to understand how and why these techniques are selected by practitioners.

There is a significant body of work proposing methods of selecting the right technique (Tiwari et al., 2012), but there are many competing approaches, any of which is only valuable if it is known by the practicing engineer and there is time to apply it. Davis and Hickey acknowledge that the literature on requirements engineering is not always used in practice, so to increase the utility of the research, practitioners must be taught when various requirements approaches work most effectively (Davis & Hickey, 2002). The findings of this study suggest that the best strategy for eliciting requirements in resource-limited new product development is a deliberate selection of techniques that need the least time, personnel, and available users. This result supports the use of Wellsandt's empirically-based selection criteria (Wellsandt et al., 2014), however, this framework is aimed at already in-use products and only includes eight techniques. These results also support applying Sutcliffe and Sawyer's framework of a technique's potential for discovering hard-to-elicit knowledge when developing new or specialized products (Sutcliffe & Sawyer, 2013).

Additional empirical research into the resource and quality related metrics of various techniques could benefit practitioners and enable more complete requirements. Simultaneously, we encourage engineers practicing RE to add an abbreviated literature review on the latest techniques and their selection to their RE process, to stay informed on optimal practices, which should improve the quality and completeness of their requirements without significant overhead.

The discrepancy that exists between the conventional literature, agile methodology doctrine, and the empirical findings of this study regarding process formality, the appropriate level of documentation, and their value in the requirements elicitation process should be resolved. The suggestion is to determine exactly when and why formal processes and complete documentation is necessary and when it is worth reducing this burden, specifically as it pertains to requirements elicitation. Similarly, there exists a dilemma that agile methods are effective when time and resources are limited, but when this coincides with the limited user access that is common in discontinuous innovation, this user-centered approach may not be the most effective. Further research examining technique selection factors could include the conditions when agile-like methods (i.e., less documentation and formality, more user access needed) are effective.

Finally, we suggest a reexamination of the resource and time constraints on SBIR-funded product development efforts with the aim of increasing requirements completeness and product success. Our findings suggest that the short timeline and minimal funding effectively encourages lean development, but, like startups, it also limits the time to access enough users and validate requirements, which risks incompleteness and limited product success. With the SBIR program's phased approach, the value placed on an artificial milestone (i.e., end of a Phase) prioritizes achieving an arbitrary benchmark over producing a successful product, which promotes an imbalanced approach focused on only functional requirements and ignoring organizational goals and market needs. To support the SBIR program's stated innovation goals, an adjustable development timeline and funding amount based on the problem scope is proposed, outlined by the developers, and evaluated by reviewers, to ensure achievable requirements completeness. Within the constraints of the current SBIR framework, it is recommended that practitioners performing PD assess the problem and domain prior to the proposal, to ensure the allocated time and funding is enough to gather complete requirements, taking into account the challenges of user access and requirements validation. Based on our findings, an appropriate development scope should improve requirements completeness and improve product success. Though not always constrained but such strict time limitations, this recommendation applies to practice with agile approaches and startups, which may not otherwise scope development to address user access issues, requirements validation, and include non-functional requirements.

3.2 Outlook

There are already many studies examining RE in agile environments that expose the challenges of conducting RE in modern product development, but additional work is needed by both the literature and practitioners to stay current on the approaches used in industry and what is most effective within the constraints of modern PD. We hope the findings of this study encourage efforts to assess and validate the conditions, effectiveness, and resource requirements of real-world practices and then update the literature that prescribes validated approaches to practitioners, to bring literature and practice into better alignment. If industry practitioners also review the updated literature prior to their RE efforts, they may benefit from applying the most effective techniques and producing more complete requirements within the challenging constraints of modern PD.

3.3 Limitations

We acknowledge there are limitations of these case studies. Though their similarities to other modern PD and agile-like development environments, like startups, should make the findings widely generalizable, the cases examined here were funded by the DOD through the SBIR program’s phased development approach, and military personnel were the primary users. The internal validity of this study may be limited by the fact that three or less engineers from each case were interviewed and biases could exist since the researcher also participated in each case. The external validity may be limited because only three cases were studied and all three were from the same company, but they were strategically selected with diverse technical problems and structural characteristics similar to other lean PD efforts, which was verified by study participants with extensive outside experience.

CHAPTER IV

CONCLUSION

4.1 Results Summary

This multi-case study summarized RE approaches and discovered patterns that both align and diverge from the literature. Like conventional RE literature outlines, elicitation was a critical upfront task, had a significant role in the product development process, involved common techniques performed by engineers, and resulted in a large set of requirements to guide development. Like other findings on RE in agile environments, we found minimal formality and documentation, mixed formal RE training, a focus on user feedback, consistent use of prototypes, incomplete requirements, and challenges regarding user availability, requirements validation, and gathering non-functional requirements. Our findings indicate that though engineers in these case studies did not explicitly use agile methods and had some amount of traditional RE knowledge, their practices and challenges closely reflect those found by studies on RE in agile development, where agile practices are already a departure from traditional methods.

Our key finding suggests an exhaustive examination of technique choice—whether general agile-like methods or specific RE techniques—by both the literature and practitioners, to ensure the most effective methods are used to produce complete requirements within the constraints of the product development. We suggest examining the conditions when agile-like methods can produce complete requirements if resources are constrained and user access is limited, given the literature and our findings show modern product development is more agile-like in practice and user unavailability is common in DNPD. We suggest further investigation into how and why RE practitioners select techniques, the development of technique metrics regarding their quality and resource requirements, and a technique selection literature review to practitioners prior to beginning RE, all with the aim of improving technique selection for better use of limited time and resources, increased adoption of up-to-date best practice by RE engineers, and ultimately more complete requirements and product success. We also advocate for a new approach to the scoping and phasing of SBIR-funded development, to better align the provided time and resources with the problem scope and complexity and reduce risk of requirements incompleteness for improved opportunity of product success.

REFERENCES

Alzahrani, A. A. H. (2020). Analyzing the Factors Impacting the Choice of the Fact-Finding Technique for Requirements Elicitation. *International Journal on Information Technologies & Security*, *12*(1), 15–36.

Anwar, F., & Razali, R. (2012). A Practical Guide to Requirements Elicitation Techniques Selection-An Empirical Study. *Middle-East Journal of Scientific Research*, *11*(8), 1059–1067.

Boehm, B. (2000, July). Requirements that Handle IKIWISI, COTS, and Rapid Change. *Software Management*, 99–102.

Carrillo De Gea, J. M., Nicols, J., Alemn, J. L. F., Toval, A., Ebert, C., & Vizcano, A. (2011). Requirements engineering tools. *IEEE Software*, *28*(4), 86–91. https://doi.org/10.1109/MS.2011.81

Christel, M., & Kang, K. (1992). Issues in requirements elicitation. *Requirements Engineering Project*, *September*, 1–77.

Crowne, M. (2002). Why software product startups fail and what to do about it. *IEEE International Engineering Management Conference*, *1*, 338–343.

Daneva, M., Van Der Veen, E., Amrit, C., Ghaisas, S., Sikkel, K., Kumar, R., Ajmeri, N., Ramteerthkar, U., & Wieringa, R. (2013). Agile requirements prioritization in large-scale outsourced system projects: An empirical study. *Journal of Systems and Software*, *86*(5), 1333–1353. https://doi.org/10.1016/j.jss.2012.12.046

Davey, B., & R. Parker, K. (2015). Requirements Elicitation Problems: A Literature Analysis. *Issues in Informing Science and Information Technology*, *12*, 071–082. https://doi.org/10.28945/2211

Davis, A. M., & Hickey, A. M. (2002). Requirements Researchers: Do We Practice What We Preach? *Requirements Engineering*, *7*(2), 107–111. https://doi.org/10.1007/s007660200007

Gershenson, J. K., & Stauffer, L. A. (1999). A Taxonomy for Design Requirements from Corporate Customers. *Research in Engineering Design*, *11*(2), 103–115.

Gralha, C., Damian, D., Wasserman, A. I., Goulão, M., & Araújo, J. (2018). The evolution of requirements practices in software startups. *Proceedings - International Conference on Software Engineering*, 823–833. https://doi.org/10.1145/3180155.3180158

Harris, H. J. (2006). *Requirements Dilemma*.

He, W., & Yan, G. (2014). Mining Blogs And Forums To Understand the Use of Social Media in Customer Co-creation. *Article in The Computer Journal*. https://doi.org/10.1093/comjnl/bxu038

Hickey, A. M., & Davis, A. M. (2003). Elicitation technique selection: How do experts do it? *Proceedings of the IEEE International Conference on Requirements Engineering*, *2003-Janua*, 169–178. https://doi.org/10.1109/ICRE.2003.1232748

Inayat, I., Salim, S. S., Marczak, S., Daneva, M., & Shamshirband, S. (2015). A systematic literature review on agile requirements engineering practices and challenges. In *Computers in Human Behavior* (Vol. 51, pp. 915–929). Elsevier Ltd. https://doi.org/10.1016/j.chb.2014.10.046

ISO/IEC/IEEE. (2015). *Systems and software engineering — System life cycle processes (15288:2015)* (Vol. 15288). International Organization for Standarization. www.iso.org

ISO/IEC/IEEE. (2018). *Systems and software engineering-Life cycle processes-Requirements engineering (29148:2018)*. www.iso.org

Janits, M. (2013). *Practical Insights into Recurring Issues of Requirements Elicitation.* Umeå University.

Kotonya, G., & Sommerville, I. (1996). Requirements Engineering with Viewpoints. *Software Engineering Journal*, 5–18. https://doi.org/10.1109/9781118156674.ch3

Kotonya, G., & Sommerville, I. (1998). Requirements Engineering - An introduction to requirements engineering. In *Architectural Design*. http://www.bth.se/fou/cuppsats.nsf/all/dfb2e7bad7b713e6c1256ee50042485b/$file/FINAL_George_Wanganga_Master Thesis _Revised 1%5B2%5D.3.1.1.3.Latest.pdf

Lee, S., & Rine, D. (2004). Case study methodology designed research in software engineering methodology validation. *Proceedings of the Sixteenth International Conference on Software Engineering and Knowledge Engineering (SEKE'04)*, 117–122. http://machinediscovery.com/nise/Papers/SEKE04LeeRineFinalCamera-header.pdf

Lindvall, M., Basili, V., Boehm, B., Costa, P., Dangle, K., Shull, F., Tesoriero, R., Williams, L., Zelkowitz, M., Mx^a, Q. [, Jgqhyqhihy, $, & Mppmeqw$gwgrgwyihy, [. (2002). Empirical Findings in Agile Methods. *LNCS*, *2418*, 197–207. http://www.softwarereality.com/lifecycle/xp/case_against_xp.jsp

Medeiros, J., Vasconcelos, A., Goulão, M., Silva, C., & Araújo, J. (2017). An approach based on design practices to specify requirements in agile projects. *Proceedings of the ACM Symposium on Applied Computing*, *Part F1280*, 1114–1121. https://doi.org/10.1145/3019612.3019753

Melegati, J., Goldman, A., Kon, F., & Wang, X. (2019). A model of requirements engineering in software startups. *Information and Software Technology*, *109*, 92–107. https://doi.org/10.1016/j.infsof.2019.02.001

Merriam, S. B. (1998). *Qualitative research and case study applications in education* (Rev. and e). San Francisco (Calif.) : Jossey-Bass. http://lib.ugent.be/catalog/rug01:000682537

Nuseibeh, B., & Easterbrook, S. (2000). *Requirements Engineering: A Roadmap*. *March 2000*.

Pacheco, C., Garcia, I., & Reyes, M. (2018). Requirements elicitation Techniques: A systematic literature review based on the maturity of the techniques. *IET Software*, *12*(4), 365–378. https://doi.org/10.1049/iet-sen.2017.0144

Paetsch, F., Eberlein, A., & Maurer, F. (2003). Requirements Engineering and Agile Software Development. *WET ICE 2003. Proceedings. Twelfth IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises, 2003.*, 1–6.

Paternoster, N., Giardino, C., Unterkalmsteiner, M., Gorschek, T., & Abrahamsson, P. (2014). Software development in startup companies: A systematic mapping study. In *Information and Software Technology* (Vol. 56, Issue 10, pp. 1200–1218). Elsevier. https://doi.org/10.1016/j.infsof.2014.04.014

Pichler, M., Rumetshofer, H., & Wahler, W. (2006). Agile requirements engineering for a social insurance for occupational risks organization: A case study. *Proceedings of the IEEE International Conference on Requirements Engineering*, 251–256. https://doi.org/10.1109/RE.2006.8

Racheva, Z., Daneva, M., & Herrmann, A. (2010). A conceptual model of client-driven agile requirements prioritization: Results of a case study. *ESEM 2010 - Proceedings of the 2010 ACM-IEEE International Symposium on Empirical Software Engineering and Measurement*. https://doi.org/10.1145/1852786.1852837

Rafiq, U., Bajwa, S. S., Wang, X., & Lunesu, I. (2017). Requirements elicitation techniques applied in software startups. *Proceedings - 43rd Euromicro Conference on Software Engineering and Advanced Applications, SEAA 2017*, 141–144. https://doi.org/10.1109/SEAA.2017.73

Ramesh, B., Cao, L., & Baskerville, R. (2007). Agile requirements engineering practices and challenges: an empirical study. *Information Systems Journal*, *20*(5), 449–480. https://doi.org/10.1111/j.1365-2575.2007.00259.x

Ries, E. (2011). *The lean startup: How today's entrepreneurs use continuous innovation to create radically successful businesses*. Currency.

Sharma, S., & Pandey, S. K. (2013). Revisiting Requirements Elicitation Techniques. In *International Journal of Computer Applications* (Vol. 75, Issue 12).

Sharma, S., & Pandey, S. K. (2014). Requirements elicitation: Issues and challenges. *2014 International Conference on Computing for Sustainable Global Development (INDIACom)*, 151–155.

Sood, V. R., & Arora, M. (2012). Comparison of Requirements Elicitation Techniques. *International Journal of Advances in Computing and Information Technology*, *1*(4), 378–387. https://doi.org/10.6088/ijacit.12.14005

Stake, R. E. (1995). The art of case study research. In *The art of case study research.* (pp. xv, 175–xv, 175). Sage Publications, Inc.

Sutcliffe, A., & Sawyer, P. (2013). Requirements elicitation: Towards the unknown unknowns. *2013 21st IEEE International Requirements Engineering Conference, RE 2013 - Proceedings*, 92–104. https://doi.org/10.1109/RE.2013.6636709

Tiwari, S., Rathore, S. S., & Gupta, A. (2012). Selecting requirement elicitation techniques for software projects. *2012 CSI 6th International Conference on Software Engineering, CONSEG 2012*. https://doi.org/10.1109/CONSEG.2012.6349486

Tripathi, N., Klotins, E., Prikladnicki, R., Oivo, M., Pompermaier, L. B., Kudakacheril, A. S., Unterkalmsteiner, M., Liukkunen, K., & Gorschek, T. (2018). An anatomy of requirements engineering in software startups using multi-vocal literature and case survey. *Journal of Systems and Software*, *146*, 130–151. https://doi.org/10.1016/j.jss.2018.08.059

Veryzer, R. W. (1998). Discontinuous Innovation and the New Product Development Process. *Journal of Product Innovation Management*, *15*(4), 304–321. https://doi.org/10.1016/s0737-6782(97)00105-7

Wellsandt, S., Hribernik, K. A., & Thoben, K. D. (2014). Qualitative comparison of requirements elicitation techniques that are used to collect feedback information about product use. *Procedia CIRP*, *21*, 212–217. https://doi.org/10.1016/j.procir.2014.03.121

Yazan, B. (2015). Three Approaches to Case Study Methods in Education: Yin, Merriam, and Stake. *The Qualitative Report*, *20*(2), 134–152. https://doi.org/10.46743/2160-3715/2015.2102

Yin, R. K. (2014). *Case Study Research: Design and Methods* (5th ed.). SAGE Publications. https://books.google.com/books?id=FzawIAdilHkC

Yin, R. K., & Davis, D. (2007). Adding new dimensions to case study evaluations: The case of evaluating comprehensive reforms. *New Directions for Evaluation*, *2007*(113), 75–93. https://doi.org/10.1002/ev.216

Zhang, Z. (2007). Effective requirements development-a comparison of requirements elicitation techniques. *Software Quality Management XV: Software Quality in the Knowledge Society, E. Berki, J. Nummenmaa, I. Sunley, M. Ross and G. Staples (Ed.) British Computer Society*, 225–240. https://www.researchgate.net/publication/228717829

Zowghi, D., & Coulin, C. (2005). Requirements Elicitation: A survey of techniques, approaches, and tools. In *Engineering and Managing Software Requirements* (pp. 19–46). https://link.springer.com/chapter/10.1007/3-540-28244-0_2

APPENDIX A

PARTICIPANT INTERVIEW QUESTIONS

A.1 Participant Interview Questions

1) Background / experience
   a) What is your background in conducting requirements elicitation?
   b) What is/are your degree(s) in?
   c) Have you been formally trained or taught RE in education setting?
   d) How many times have you performed RE?
   e) Have you performed RE at other companies or different types of projects?
   f) How many years have you been an engineer/scientist performing RE?
2) Project roles
   a) What was your role in Project [S/T/W]?
   b) How many people performed RE for this project?
3) Process: Please answer the following specifically for Project X
   a) Can you describe how you performed requirements elicitation?
   b) What techniques would you say you used?
   c) Describe the timing and scope of the elicitation process
   d) What research did you conduct to determine the users or markets for the product?
   e) What research did you conduct to determine prerequisite requirements: technical specifications, features etc?
   f) How did you find users/experts to interview?
   g) Describe your interview process
   h) What was the consensus/consistency of interview responses?
   i) Was user feedback limited to the interview or was it iterative/continuous
   j) Did you use prototypes or props to gather feedback? If yes, was it effective?
   k) How did you document the requirements?
   l) How did you determine when to stop eliciting requirements, or when they were complete?
   m) How did you deal with conflicting requirements (i.e., two users provided different feedback)?
4) Challenges
   a) What challenges did you face during elicitation?
   b) Did you run into any issues finding/contacting/getting responses from users/customers/experts?
   c) What reasons do you believe affected user availability?
   d) What reasons do you believe affected user responsiveness, feedback, or helpfulness?
   e) What effect did the project timeline have on the elicitation process? (i.e., did it make the process more challenging?)
   f) What effect did the number of developers/researchers have on the elicitation process?
   g) What effect did the funding have on the elicitation process?
   h) What would you change about the process or do differently next time?

APPENDIX B

PARTICIPANT SURVEY

B.1 Participant Survey

5) How important do you view RE in the PD process?
   a) Description
   b) 1-5 (5 very important)
6) How well was the elicitation process performed?
   a) Description
   b) 1-5 (5 very well)
7) In your opinion, how complete were the requirements?
   a) Description
   b) 1-5 scale (5 complete, 3 partly complete, 1 incomplete)
8) How much did the requirements' completeness affect the project result?
   a) Description
   b) 1-5 scale (5 large effect, 1 no effect)
9) How challenging was eliciting requirements from this user population?
   a) Description
   b) 1-5 (5 very challenging)
10) How unique do you think the elicitation challenges were to this problem/project?
   a) Description
   b) 1-5 (5 Unique / 1 common)