Journal of Engineering Technology and Applied Physics

Evolution of Requirements Engineering in Agile Methodology – Literature Review

Ayesha Anees Zaveri[1,*], Juliana Jaafar[2], Eiad Yafi[3] and Sarama Shehmir[4]
[1,2]*Malaysian Institute of Information Technology, Universiti Kuala Lumpur, 50250 Kuala Lumpur, Malaysia.*
[3]*Faculty of Engineering and Information Technology, University of Technology Sydney, NSW 2007, Sydney, Australia.*
[4]*Department of Electrical and Computer Engineering, Toronto Metropolitan University, Toronto, Ontario, Canada.*
**Corresponding author*: Zaveri.ayesha@s.unikl.edu.my, *ORCiD*: 0009-0007-7230-1696
https://doi.org/10.33093/jetap.2024.6.2.9
Manuscript Received: 31 March 2024, Accepted: 29 April 2024, Published: 15 September 2024

***Abstract* — Requirements Agile approaches have transformed engineering. This paper shows how RE in Agile software development has evolved from documentation-heavy to collaborative, adaptable, and customer-focused. Agile was born in the mid-1990s when the industry realized it needed to respond to changing client needs and market volatility. This evolution includes iterative development, client interaction, and emphasizing communication above documentation, as discussed in the paper. By comparing conventional and Agile RE approaches, we demonstrate the benefits of adapting to change, working with customers, and delivering functional software faster. This analysis provides a persuasive description of Agile RE implementation methodologies and resources through a detailed literature review and real-world experiences. User stories and backlog refinement are notable techniques. The research finishes by exploring how these techniques affect team dynamics, project success, and customer satisfaction. RE's Agile difficulties and opportunities are also examined. The findings illuminate RE methods' successful adaptation to Agile projects' dynamic character. Software development is more responsive and effective due to this adaptation.**

Keywords*— *Agile, Agile Methodology, Requirements Engineering, Process Models, Agile Software Development.

I. INTRODUCTION

Agile requirements engineering methodology has grown significantly since the mid-1990s, with consultants playing a crucial role in its development and evolution. The methods discussed here were developed to address the demand for greater flexibility and adaptability in software development processes, specifically concerning evolving requirements and customer expectations. Various agile methodologies have been designed to accommodate the increasing frequency of changes in software requirements. These methodologies include Adaptive Software Development, Crystal, Dynamic Systems Development Method, Extreme Programming, Feature-Driven Development, Pragmatic Programming, and Scrum. Agile methodologies focus on valuing skilled individuals and their relationships in software development rather than relying heavily on detailed planning, strict processes, and extensive reuse. Shifting the focus allows more room for collaboration and a remarkable ability to adapt to customers' ever-changing needs. This ultimately leads to deliverables better suited to meet their specific requirements. There has been a noticeable shift towards emphasizing collaboration and responsiveness to customers' ever-changing needs throughout the evolution of requirements engineering in agile methodologies. The changing landscape has resulted in the implementation of various practices, including continuous improvement, development centred around customer needs, and active customer participation in the requirements-gathering process. In addition, there has been a noticeable transition from conventional requirements documentation to more dynamic and iterative methods, including user stories, backlog refinement, and feature prioritization. In recent years, the field of requirements engineering has undergone significant changes due to the adoption of agile methodology. This shift has resulted in a more flexible and customer-focused approach, where requirements are continuously reviewed and modified based on customer input and changing market dynamics. Overall, the evolution of requirements engineering in agile methodology emphasizes the importance of collaboration, responsiveness to changing customer needs, and continuous improvement. The current shift in the industry has

resulted in the implementation of various practices that aim to enhance overall performance. These practices include continuous improvement, a focus on customer needs during development, and early and frequent involvement of customers in the requirements-gathering process.

Agile software development (ASD) is a methodology that offers advantages such as timely project completion and enhanced customer satisfaction. Its primary objective is to deliver corporate value through the iterative delivery of software solutions. Hence, the development process is executed gradually and evidence-based, conferring the benefit of promptly altering the product development path. Moreover, these techniques revolve around the central role of humans and their interactions [1].

As shown in Fig. 1, Agile Software Development (ASD) is frequently employed in contexts characterized by the need to address elaborate adaptive challenges. Agile techniques, such as Scrum, Kanban, and Extreme Programming (XP), are frequently integrated with Human-Centred Design (HCD) activities in developed applications. This integration of approaches aims to create a value-driven organization wherein systems may be designed and developed to meet user requirements effectively by providing a favourable user experience (UX) [2].

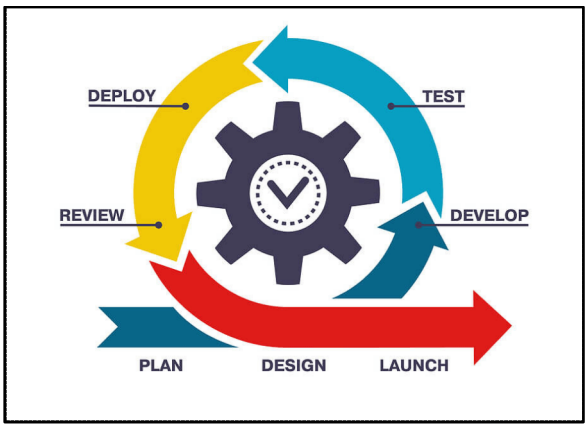

Fig. 1. Agile Software Development.

Requirements engineering refers to the systematic and structured approach of identifying and capturing stakeholders' requirements and desires and transforming them into a comprehensive and mutually agreed-upon set of precise requirements. These requirements serve as a fundamental foundation for all subsequent development endeavours. The primary objective of requirements engineering approaches is to establish a complete and definite problem statement while ensuring the proposed solution's correctness, reasonability, and effectiveness [2].

Requirements Engineering (RE) in the agile methodology is a dynamic and iterative procedure encompassing customers' active participation, ongoing planning, reevaluating requirements' priority, and validating these requirements through incremental product delivery [3].

In the industrial sector, new trends are rapidly emerging, accompanied by the volatility of agile approaches and technologies. The ongoing enhancement of the environment contributes to a dynamic and fast-evolving knowledge base within the realm of study - the proliferation of agile methods within the industrial sector results in shifts in organizations' value systems. The emphasis on meeting user wants and delivering value has become a crucial component in product development, driven mainly by the escalating levels of competition across several domains [3].

II. BACKGROUND

Until the 2000s, software development projects were predominantly handled using traditional methodologies. This implies that initiatives were meticulously designed, precisely defined, and systematically organized, resembling the approach taken in architectural projects. In contrast to this prevailing tendency, a small group of visionary IT professionals convened in 2001 to formulate the Agile Manifesto. The experts expressed dissatisfaction with the rigidity and limitations of traditional project management methodologies, which are based on certain assumptions. These assumptions include the belief that software projects can be thoroughly planned from the outset, that project requirements remain unchanged throughout the project, and that identifying and analyzing key stakeholders and business cases can be accurately conducted at the project's inception. In essence, the conventional approach to management posits that once a project transitions into the implementation phase, all crucial aspects have been definitively defined, effectively conveyed, and are not subject to alteration. Moreover, it expects that the project team will execute all tasks precisely as outlined in the initial plan. In practice, this scenario is applicable only in a limited number of instances, primarily for initiatives characterized by minimal complexity [4].

The origins of the Agile technique may be traced back to the 1990s, a period characterized by notable changes in software development practices. During this period, numerous teams experienced dissatisfaction with the constraints imposed by conventional, plan-driven methodologies, which frequently led to the non-fulfilment of deadlines, unsuccessful projects, and dissatisfied clientele. The Agile methodology is a project management and software development technique characterized by its iterative and collaborative nature. The concepts above prioritize humans and interactions, functional software development, active customer engagement, and adaptability in response to change.

In contrast to conventional techniques, Agile strongly emphasizes flexibility, adaptability, and continuous improvement. The approach encourages extensive collaboration among cross-functional teams,

regular solicitation of customer feedback, and the capacity to adapt project needs and priorities promptly.

The Agile Manifesto has served as a catalyst for developing diverse Agile approaches and frameworks, including Scrum, Extreme Programming (XP), and Kanban, among others. The approaches exhibited a set of shared concepts and practices, encompassing iterative and incremental development, forming self-organizing teams, regular customer engagement, and the ongoing provision of feedback [5].

The Agile Manifesto, as shown in Fig. 2, was officially introduced in 2001, with the participation of 17 technologists who played a crucial role in its drafting. Four key concepts were established for agile project management to guide teams in pursuing enhanced software development.

1. The prioritization of individuals and relationships over processes and tools
2. The prioritization of functional software development over extensive documentation.
3. The prioritization of customer participation over contract negotiation.
4. fcAdapting to change rather than adhering strictly to a predetermined plan [6, 7].

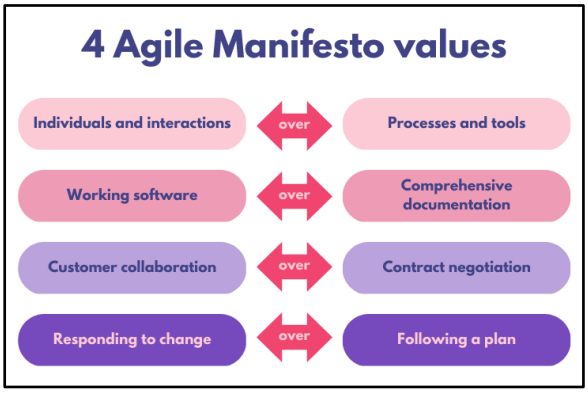

Fig. 2. The Agile Manifesto.

It is argued that a linear product development approach is unsuitable due to its limited flexibility. During the 1990s, various movements emerged in the field of software development, focusing on lightweight process models such as Scrum [7, 8], Extreme Programming (XP) [9], and Feature-Driven Development [10]. In 2001, the respective leaders of these distinct movements convened to identify shared objectives and establish a unified framework for their collective endeavours. Consequently, the Agile Manifesto was formulated [10]. The agile manifesto consists of values supported by 12 principles, as outlined in the work [11].

Despite being formulated in 2001, the Agile Manifesto is a guiding framework for agile teams. Furthermore, many of its concepts remain highly relevant within the contemporary agile community [9-11].

Agile requirement engineering has few methods. Agile Modelling is a method for creating design models for documentation. Design models clarify project requirements. The first is feature-driven development, where team members list and prioritize features. Weekly 30-minute discussions evaluate each feature. Third, the Dynamic Systems Development Method assesses all features and produces a feasibility analysis, defining its need. User engagement and periodic delivery are its primary goals.

Extreme Programming, a popular agile method, uses small iterations and user feedback. Finally, Scrum uses sprints and backlogs. A daily 15-minute meeting tracks task progress and discusses today's tasks. Scrum is "empirical process control". Therefore, project progress may be seen after each sprint. Crystal approaches are chosen from various methods for each project. In addition, Adaptive Software Development incorporates modest, incremental, change-tolerant cycles with customers [11, 12].

Requirements are the fundamental elements upon which all software products are built. As a result, Requirements Engineering (RE) assumes a crucial role in the system development process. Requirements Engineering was previously employed during the 1970s [13].

Subsequently, Requirements Engineering (RE) had a surge in popularity with the inception of its inaugural conferences in the 1990s. The waterfall model, first proposed by Royce in 1970, as shown in Fig. 2, is an example of a traditional process model that follows a sequential approach and begins with an initial design phase. Figure 2 illustrates the consecutive steps that underlie the phenomenon. At the onset of these initiatives, all requirements are synthesized and documented in a specification report. Based on the available information, the timetable and budget are estimated. The software development process involves iterations, but he also emphasized a specific point at which the requirements analysis is complete. Within the industrial context, it is commonly assumed that requirements are factual and that any modifications are implemented through a rigorous change request procedure, typically adhering to conventional process models [14, 15].

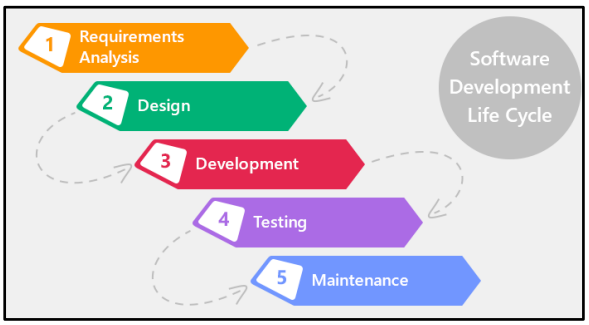

Fig. 3. Sequential phases approach to software development.

Requirements Engineering (RE) encompasses a comprehensive range of operations that involve identifying, documenting, and managing requirements. Employing systematic and replicable

methodologies is recommended to guarantee product requirements' comprehensiveness, coherence, and pertinence. The requirements are compiled and documented in a specification document, incorporating additional details as the product development process progresses [15].

The primary emphasis in the existing literature on Agile Requirements Engineering (RE) centres around the active participation of stakeholders and users. The scholarly literature investigates a range of approaches, procedures, and techniques that can be employed to actively engage stakeholders and users in the Agile Requirements Engineering (RE) process. This encompasses comprehending the methods for eliciting and prioritizing requirements, engaging stakeholders in the decision-making process, and ensuring that user wants and preferences are sufficiently considered.

The literature review also investigates the difficulties and recommended approaches for incorporating Agile techniques into requirements engineering practices. This encompasses the examination of the scalability and application of Agile Requirements Engineering (RE) in vast and complicated projects. It also involves effectively managing requirements in geographically dispersed teams and handling needs that evolve in a dynamic environment. Additionally, it entails the establishment of traceability and consistency in Agile RE projects conducted on a large scale.

Moreover, the literature review emphasizes the necessity for empirical investigations that assess the efficacy and consequences of Agile requirements engineering practices. Numerous scholarly investigations put forth conceptual frameworks, models, and recommendations. Yet, there is a need for empirical data to substantiate the tangible advantages and results derived from the implementation of Agile Requirements Engineering (RE) in practical project settings. Hence, the primary objective of the literature review is to address this disparity by offering empirical observations and practical ramifications for Agile Requirements Engineering (RE).

In general, the existing body of research in Agile Requirements Engineering mostly centres around the active participation of stakeholders and users, tackling various issues, recommending optimal approaches, and presenting empirical findings to enrich the comprehension and implementation of Agile methodology within requirements engineering procedures [14-16].

A study found that requirement phases cause 37% of project issues. Requirement engineering should be prioritized during project development. When to do requirement engineering in agile and traditional development differs.

Agile development requirements and engineering techniques vary in each project. Agile requirement engineering includes Feasibility Study, Requirement Elicitation, Analysis, Documentation, Validation, and Management [17, 18].

First, a feasibility study gathers project details. It also assesses if the organization can handle the project and whether it has resources. Next, elicitation needs to involve stakeholders and agile team members discussing system features. Interviewing the customer, brainstorming project ideas, using case analysis, and using ethnography are used. Finding out how people work is done through observation. Another demand-collecting method is focus groups, where 4-9 people from varied backgrounds debate project features [17, 18].

Requirement Analysis also resolves confusing or contradictory requirements. Some strategies are Joint Application Department, Modelling, and Prioritisation. The Joint Application Department is a workshop that promotes conflict resolution among stakeholders during project development. Modelling shortens the design-analysis gap. It shows the project status visually. Prioritization assesses essential requirements to develop them first. Agile prioritizes requirements, implementing new and old requirements based on their priority.

In addition, Requirement Documentation provides essential project documentation. Two team members document the storyboard and use case features for ongoing documentation. However, more documentation can help long-term projects. User stories document requirements well. According to [4], user stories define requirements and estimate work succinctly. User stories should be Specific, Measurable, Achievable, Relevant, and Time-bound.

Requirement Validation also includes Acceptance Testing to ensure the system meets user needs. Required Reviews, Unit Testing, and Evolutionary Prototyping are included. Evolutionary prototyping adds features in order of priority using user input. Lastly, requirement management manages stakeholder and project team communication [19].

Agile methodologies have demonstrated higher success than conventional requirement engineering practices. In the agile method, it is possible to modify project requirements at any point during its lifecycle. In contrast, in traditional requirements engineering, gathering requirements occurs at the outset, and any further changes introduce a disconnect between the documentation and the development process. According to a poll done among professionals who have firsthand experience with both techniques, the findings indicate that agile methodology has more remarkable performance in terms of flexibility, fostering strong customer relationships, and its ability to adapt to change [20].

Requirement Engineering in Agile projects involves a diverse range of tasks and methodologies. This study presents a comprehensive literature review on several Requirements Engineering (RE) methodologies. It offers a critical critique of these strategies. Additionally, various methods have been suggested for agile project requirements engineering (RE). Using automated tools and procedures in agile methodologies can enhance the effectiveness of

requirements engineering (RE). Selecting a suitable and previously experienced RE approach is paramount in comparable scenarios. User engagement, observation, and ethnography are more effective methodologies for agile projects prioritizing rapid software delivery. According to the survey reported [21], the combination of several strategies has the potential to yield effective outcomes. Prototyping techniques have been found to enhance project resources. However, their applicability is constrained primarily to small to medium-sized projects [21].

One technique considered in the literature is crowd-based requirement gathering. This process entails the participation of people from diverse backgrounds and age cohorts to collect project needs. The prioritization of work and resolution of problems are typically addressed through voting or engaging in talks with a group of individuals. According to a survey conducted [22], it has been determined that utilizing crowd-based criteria enhances originality and diversity. Project teams play a crucial role in the field of requirement engineering.

III. PROBLEM STATEMENT

The typical or traditional Software Development methodology is characterized by a linear progression, wherein each process phase follows a sequential order. The chosen method is contingent upon reliable instruments and consistent expertise. Every project has a standardized life cycle encompassing many stages: feasibility, planning, design, construction, testing, production, and support. The project is comprehensively planned, with no allowance for modifications to the needs. This technique operates under the assumption that time and cost are dynamic variables while requirements remain constant. The abovementioned factor is the underlying cause of traditional project management's budgetary and temporal challenges [22].

In contrast to traditional systems that prioritize upfront planning and emphasize cost, scope, and time, Agile management emphasizes cooperation, customer collaboration, and adaptability. The Agile approach dismisses conventional project management approaches due to their perceived drawbacks of being burdensome, constraining, and ill-suited for the contemporary period characterized by rapidity. Agile project management is characterized by an iterative approach that seeks to consistently integrate user feedback and achieve continuous releases across each iteration of the software development project, as depicted in the provided Figure. Each task output can be considered as a product that is being offered to stakeholders. The design of teams and work structures is centred on developing products or services that directly use customers or clients [23].

In agile methodologies, requirements engineering (RE) is a dynamic and iterative process in which requirements evolve throughout the agile iterations. This stands in contrast to the waterfall approach, where RE is typically finalized before deployment, and any modifications to requirements are usually negotiated with the client through a formal change management process. Some conventional issues associated with requirements engineering are reduced customer engagement, excessive scope definition, challenges in requirements documentation, and communication difficulties [22, 23].

The existing body of literature about Agile Requirements Engineering (RE) has made notable advancements in comprehending the amalgamation of Agile techniques with Human-Centred Design (HCD) and recognizing the associated issues and practices. Nevertheless, there still needs to be more in the existing body of scholarly work that necessitates further investigation and resolution.

A significant deficiency exists in prioritizing stakeholder and user engagement within Agile requirements engineering. Although several studies have briefly addressed this topic, there exists a requirement for a more extensive study that delves into the precise methods, tactics, and tools employed to successfully engage stakeholders and users throughout the Agile requirements engineering (RE) process. This encompasses comprehending the methods for eliciting and prioritizing requirements, engaging stakeholders in the decision-making process, and ensuring that user wants and preferences are sufficiently considered [23, 24].

Another area for improvement lies in the restricted emphasis on the scalability and application of Agile Requirements Engineering (RE) in large and complicated projects. Most extant scholarly works have predominantly concentrated on projects of limited scope. Consequently, there is a requirement for intellectual inquiry that examines the difficulties and optimal approaches for implementing Agile Requirements Engineering (RE) in more extensive and intricate environments. This encompasses proficiently managing requirements in distributed teams, addressing the evolution of needs in a dynamic context, and ensuring traceability and consistency in large-scale Agile requirements engineering projects.

Additionally, it is imperative to conduct further empirical research to assess the efficacy and consequences of Agile requirements engineering practices. Numerous scholarly investigations have put forth various frameworks, models, and Agile Requirements Engineering (RE) principles. However, more empirical data is needed to substantiate the tangible advantages and consequences resulting from the practical implementation of Agile RE in real-life projects. Further empirical research is required to confirm the suggested methodologies and offer valuable insights into the practical ramifications of Agile Requirements Engineering.

Addressing these deficiencies in the existing body of knowledge will enhance the overall comprehension of Agile requirements engineering (RE) and furnish practitioners with essential insights and recommendations for efficiently implementing Agile methodology in requirements engineering procedures. [18, 25].

IV. REQUIREMENTS EVOLUTION IN AGILE AND ITS ROLE IN AGILE PROJECTS

Requirements Engineering is crucial in Agile projects, serving as the basis for comprehending and handling customer needs and expectations.

The development of requirements engineering in Agile methodology represents a significant shift from traditional, inflexible software development approaches to a more adaptable and iterative process. This shift in approach can be traced back to the mid-1990s when lightweight Agile methods started to gain popularity as a response to the more rigid waterfall-oriented methods. The critical components of this transformation in requirements engineering encompass:

A. Flexibility and Adaptability: Agile methodologies, such as Scrum and Kanban, were designed to easily accommodate and adapt to evolving software requirements, even during the later stages of development.

B. Customer Focus: Agile methodologies don't just prioritize customer collaboration over contract negotiation. They put the customer at the heart of the development process. This approach ensures continuous feedback and guarantees that the final product is tailored to their needs, making stakeholders feel empowered and integral to the project's success.

C. Iterative Development: Agile methodologies revolutionized the software development process by introducing planning, development, and evaluation cycles. This fosters a culture of ongoing improvement and flexibility in the face of evolving circumstances.

D. Simplicity and Efficiency: Agile methods strongly emphasize delivering functional software in a timely and efficient manner, focusing on minimizing unnecessary work and documentation. The main goal is to maximize the value of the software being developed.

E. Cross-functional Teams: Collaboration is critical in Agile requirements engineering. It's all about self-organizing teams, where software developers and project managers are crucial, working closely with stakeholders to get the job done. This guarantees a more profound comprehension of requirements and a quicker adaptation to change, empowering them in the process.

F. Continuous Improvement: Agile methodologies don't just promote a constant enhancement culture. They inspire it. Teams are encouraged to frequently analyze their workflows to discover methods for improving efficiency, fostering a sense of inspiration and motivation for ongoing enhancement.

G. Interactive Techniques: The focus has shifted from extensive documentation to embracing interactive and customer-centric practices like user stories, backlog grooming, and feature prioritization. This shift has fostered a more collaborative approach to shaping and enhancing requirements [18, 24, 25]

The primary areas of emphasis in Agile Methods in Requirements Engineering (RE) include the customer, waste in requirements, requirements evolution, and non-functional requirements.

In agile methods, requirements engineering emphasizes the involvement of customers and stakeholders. The interaction between the development team and stakeholders is characterized by its complexity, primarily stemming from the diverse perspectives held by stakeholders regarding the problem at hand. Agile methods address the challenge of managing multiple stakeholders by consolidating their representation into a single individual who acts on behalf of all stakeholders throughout the project. The customer needs domain expertise and the ability to make critical decisions, including product acceptance and prioritizing requirements [25, 26].

The evolution of requirements in Agile methods assumes that gathering all user requirements at the outset of a development project is challenging. Additionally, these methods acknowledge that requirements are subject to change over time due to customer preferences or shifts in the technical and socio-economic landscape. Agile organizations understand that the occurrence of changes is unavoidable. As such, they incorporate the management of variability into their development process. It is acknowledged that the initial stages of a project often lack a comprehensive understanding of the requirements. The notion of requirements changes. The act of making alterations does not incur significant expenses. Agile methodologies operate under the assumption that the cost associated with implementing modifications in a product remains relatively stable throughout its lifespan; nevertheless, this assumption may not hold in all circumstances. The expenses related to implementing changes tend to increase dramatically as time progresses. However, when development phases are consolidated into brief iterations and crucial choices are postponed until the last feasible moment, the escalation of costs is constrained. Agile techniques employ a contractual arrangement known as a variable scope-variable pricing contract to handle requirements evolution effectively. This implies that the system's incorporated features and associated costs undergo evolution. The project involves ongoing requirements negotiation between the customer and the development team [27, 28].

The management of variability is a crucial aspect in various fields and industries. It involves identifying, analyzing, and controlling factors that clarify the necessary implementation tasks and render the sequence of their execution inconsequential. The requirements must exhibit a high degree of independence.

At the commencement of each iteration, an activity is conducted to collect and prioritize needs. During this process, novel requirements are recognized and assigned priority. This methodology facilitates the

identification of the most crucial elements inside the current project. Features are mainly introduced based on their prioritization rather than their functional interdependence.

Non-functional requirements refer to the aspects of a system or software that are not directly related to its functionality. Agile methodologies need a universally agreed approach for eliciting and maintaining non-functional needs. These requirements are typically gathered implicitly during the process of collecting requirements. The significance of outlining non-functional needs is comparatively less in this context owing to the ongoing engagement with the customer.

Utilizing tools for requirements management in Agile methods is also a part of this methodology. The essential tools for the task are paper, pencil, and pinboards for visualizing ideas and concepts. Additionally, UML modelling tools are necessary for creating and representing system models. Requirements negotiation tools facilitate effective communication and agreement on project requirements. Lastly, instant messaging applications are recommended for seamless and efficient communication among team members [26-28].

V. METHODOLOGIES AND THEIR GAPS

Within the realm of research background, one can encounter various studies that put forth process models about the agile requirements engineering (RE) domain. These process models exhibit the ongoing management of requirements through the active participation of stakeholders and users. The most pertinent ones are highlighted here.

The research proposed developing a process model called cross-discipline User Interface and Software Engineering (CRUISER) based on the principles of Extreme Programming (XP). The method commences with an Initial Requirements Up-front Phase (IRUP), which yields agile models that articulate user requirements using agile techniques such as fundamental use cases, scenarios, and prototypes. The collected data is analyzed and processed across the various stages of the CRUISER framework [26].

A case study was undertaken to examine the level of user and consumer engagement in the context of Agile Software Development (ASD). The author does not explicitly assert the proposition of a process model. Still, the study's findings demonstrate the presence of an implicitly applied process model. Kautz incorporates Participatory Design activities within Extreme Programming (XP). The agile team can identify issues related to misunderstandings of requirements at an early stage, preventing them from escalating into more significant problems. This is achieved by the involvement of an onsite customer and the regular review process with users and customers [27].

The research proposed a metamodel for artefact-based requirements engineering (RE) that encompasses the entire RE domain without focusing on any development approach. The metamodel is derived from two established requirements engineering models utilized in the industry. On the one hand, it offers a valuable, comprehensive examination of managing artefacts in requirements engineering. In contrast, using the metamodel facilitates the development of RE process models based on enterprises' artefacts [28].

Another research proposed the HCD (Human-Centred Design) process encompasses several sequential processes. These steps include planning the HCD process, comprehending and specifying the design's context, articulating user requirements, generating design solutions that align with these requirements, and assessing the designs against the established criteria. The author proposes a series of agile methodologies that can be employed at every stage. Furthermore, the author suggests certain artefacts produced while implementing agile methods [29].

The research introduced the process model called Mockup-Driven Development (MockupDD). The strategy employed by the authors is aligned with the principles of Model-Driven Web Engineering (MDWE). It has been seamlessly integrated with the Scrum methodology. At the onset of MockupDD, an initial phase of expeditious requirements collecting is conducted, yielding a collection of user stories. Based on this, customers and users generate visual representations, known as mock-ups, to depict these user stories. These mock-ups serve as the basis for the subsequent modelling procedure [30].

The process model developed in the research is derived from a conceptual model called Qualitative/quantitative Customer-driven Development. The importance of integrating qualitative consumer feedback throughout the initial phases of development alongside quantitative observations in subsequent stages is emphasized. The approach by Olsson et al. involves regarding requirements as hypotheses that undergo validation with customers before the commencement of development. Hypotheses are formed based on corporate strategy, innovation activities, customer input, and continuous validation cycles.

Upon analyzing the shared characteristics of the approaches above, it can be deduced that the studies conducted by Memmel *et al.* (2007), Kautz (2010), Maguire (2013), Rivero *et al.* (2014), and Olsson *et al.* (2015) all encompass process models that delineate the modus operandi within the realm of agile requirements engineering (RE). These models comprise workflows, role descriptions, and agile techniques, elucidating how work is conducted in this context [31].

The metamodel developed by Méndez Fernández *et al.* (2010) for artifact-oriented requirements engineering (RE) is designed to address RE in a broad sense rather than being specifically customized to the unique requirements of Agile Software Development (ASD). Furthermore, their focus is on an artifact-centred approach. In contrast, our research fosters

individual collaboration, adopting a more human-centred perspective.

Agile requirements engineering (RE) practices and approaches enable agile RE patterns. This research presented agile requirements engineering patterns. Agile requirements engineering patterns were developed using a three-phase pattern mining process. An agile requirements engineering pattern includes a problem and an appropriate agile way to solve it. This mapping produced 41 agile requirements engineering patterns. a) Evaluation and testing evaluate a product or system for quality, functionality, and performance. b) A story map organizes and prioritizes user stories or needs in product development. It gives a complete picture of the goods. c) Product owners represent stakeholders and customers in product development. They define and prioritize product requirements, manage the product backlog, and ensure it meets user demands [32].

Continuous requirements management involves recognizing, documenting, assessing, and prioritizing stakeholder needs and expectations throughout a project's lifecycle. Agile requirements engineering solves this problem. Continuous requirement management using instruments. This research implements the agile requirements engineering metamodel in a Scrum-oriented methodology for developing e-government web applications. Agile requirements engineering was highlighted in this situation. Agile requirements engineering solves technical or functional dependencies with other teams [32, 33].

However, the existing literature needs to offer comprehensive concepts at a higher level of abstraction. The significance of these generic principles in the contemporary business landscape lies in their application across various process models utilized by firms for diverse teams. This phenomenon results in an organization's heightened intricacy, mainly when scaled or teams employ sequential methodologies such as the waterfall model. To achieve this objective, the present study introduces a metamodel for agile requirements engineering (RE) to effectively manage the inherent complexity. Based on the available information, this metamodel for agile requirements engineering is the first of its kind [34, 35].

VI. CONCLUSION

To summarise, the ongoing enhancement of Agile methodologies through focused investigation offers the potential for substantial advantages to both practitioners and scholars. Practitioners can benefit from methods proven via empirical evidence, Agile frameworks tailored to meet the specific needs of different organizations, and advanced tools that make the Agile process more efficient and effective. However, academics can enhance their comprehension of Agile approaches, make valuable contributions to the existing knowledge, and stimulate innovation in the sector. Researchers can tackle the changing difficulties practitioners face by prioritizing empirical studies, doing cross-disciplinary research, and creating new tools and measurements. The collaboration between practice and research will create an environment where Agile methodologies can adjust to the evolving software development field and continue to be leaders in providing exceptional value to customers and stakeholders in a complex and dynamic world [36].

Abbreviations and Acronyms

ASD – Agile Software Development.

RE – Requirements Engineering.

SE – Software Engineering.

SDLC – Software Development Life Cycle

ACKNOWLEDGEMENT

I'd like to thank my university, UniKL, for its support in our research. Without our parents, family, and friends' support and patience, no attempt can succeed. Thanks to Dr Juliana Jaafar and Dr Eiad for their unwavering support and guidance while writing this paper. I appreciate everyone who helped me research this paper.

REFERENCES

[1] E. Schön, J. Sedeño, M. Mejías, J. Thomaschewski and M. Escalona, "A Metamodel for Agile Requirements Engineering," *J. Comput. and Commun.*, vol. 7, pp. 1-22, 2013.

[2] N. Saher, F. Baharom and R. Romli, "Guideline for the Selection of Requirement Prioritization Techniques in Agile Software Development: Empirical Research," *Int. J. Recent Technol. and Eng.*, vol. 8, no. 5, pp. 3381-3388, 2020.

[3] "Agile Done Right Eliminates The Need for Classical Requirements Engineering," https://www.scrum.org/resources/blog/agile-done-right-eliminates-need-classical-requirements-engineering. [10 November 2022]

[4] A. Muhammad, A. Siddique, M. Mubassher, A. Aldweesh and Q. Naveed, "Prioritizing Non-functional Requirements in Agile Process using Multi Criteria Decision Making Analysis," *IEEE Access*, vol. 11, pp. 24631-24654, 2023.

[5] "Classical Project Management vs Agile Project Management," https://www.visual-paradigm.com/scrum/classical-vs-agile-project-management/. [2023]

[6] E. Schön, "A Framework for Modeling and Improving Agile Requirements Engineering, Computer Languages and Systems Department University of Seville," PhD Thesis, 2017.

[7] V. Gaikwad and P. Joeg, "A Case Study in Requirements Engineering in Context of Agile," *Int. J. Appl. Eng. Res.*, vol. 12, no. 8, pp. 1697-1702, 2017.

[8] T. Hirotaka and I. Nonaka, "The New Product Development Game," *Harvard Busine. Rev.*, pp. 137-146, 1986.

[9] K. Schwaber, "SCRUM Development Process," in J. Sutherland, C. Casanave, J. Miller, P. Patel and G. Hollowell, *Business Object Design and Implementation*, Springer, London, 1997.

[10] K. Schwaber, "Agile Project Management with Scrum," *Microsoft Press*, 2004.

[11] K. Beck and C. Wilson, "Development of Affective Organizational Commitment: A Cross-Sequential Examination of Change with Tenure," *J. Vocation. Behavior*, vol. 56, pp. 114-136, 2000.

[12] W. Laurie, R. Balasubramaniam, C. Alistair, L. Kalle and A. Pekka, "Agile Software Development Methods: When and Why Do They Work?" in *International Federation for Information Processing Digital Library, Business Agility and Information Technology Diffusion*, 2005.

[13] L. Williams, "What Agile Teams Think of Agile Principles," *Communications*, vol. 55, pp. 71-76, 2012.

[14] E. Schön, M. J. Escalona and T. Jörg, "Agile Values and Their Implementation in Practice," *Int. J. Artif. Intellig. and Interac. Multimedia*, vol. 3, no. 5, pp. 61-66, 2015.
[15] I. Sacolick, "A Brief History of The Agile Methodology," InfoWorld, https://www.infoworld.com/article/3655646/a-brief-history-of-the-agile-methodology.html. [8 April 2022]
[16] M. D. Richter, J. D. Mason, M. W. Alford, I. F. Burns and H. A. Helton, "Software Requirements Engineering Methodology," *Technical report, DTIC Document*, 1976.
[17] W. W. Royce, "Managing the Development of Large Software Systems," in *Proc. IEEE WESCON*, vol. 26, pp. 328-388, 1970.
[18] L. Andrea De and A. Qusef, "Requirements Engineering in Agile Software Development," *J. Emerg. Technol. in Web Intellig.*, vol. 2, no. 3, pp. 212-220, 2010.
[19] F. Paetsch, A. Eberlein and F. Maurer, "Requirements Engineering and Agile Software Development," in *Twelfth IEEE Int. Workshops on Enabling Technol.: Infrastruct. for Collab. Enterpris.*, Linz, Austria, pp. 308–313, 2003.
[20] T. Fatima and W. Mahmood, "Requirement Engineering in Agile," *IJ Educat. and Manage. Eng.*, vol. 4, pp. 20-33, 2019.
[21] A. Batool, Y. Motla, B. Hamid, S. Asghar, M. Riaz, M. Mukhtar and M. Ahmed, "Comparative Study of Traditional Requirement Engineering and Agile Requirement Engineering," in *15th Int. Conf. Adv. Commun. Technol.*, pp. 1006-1014, 2013.
[22] R. Tousif Ur, M. N. A. Khan and N. Riaz, "Analysis of Requirement Engineering Processes, Tools/Techniques and Methodologies," *Int. J. Inform. Technol. and Comput. Sci.*, vol. 5, pp. 40-48, 2013.
[23] G. Umesauda, M. Murad and W. Mahmood, "Crowd-Based Requirement Engineering," *Int. J. Educ. and Manage. Eng.*, vol. 8, no. 3, pp. 43-53, 2018.
[24] D. Turk, R. France and B. Rumpe, "Assumptions Underlying Agile Software Development Processes," *J. Database Manage.*, vol. 16, no. 4, pp. 62-87, 2005.
[25] A. Rasheed, B. Zafar, T. Shehryar, N. A. Aslam, M. Sajid, N. Ali, S. H. Dar and S. Khalid, "Requirement Engineering Challenges in Agile Software Development," *Math. Probl. in Eng.*, vol. 2021, pp. 696695, 2021.
[26] A. Sillitti and G. Succi, "Requirements Engineering for Agile Methods," *Engineering and Managing Software Requirements*, pp. 309-326, Springer, 2005.
[27] T. Memmel, F. Gundelsweiler and H. Reiterer, "Agile Human-Centered Software Engineering," in *21st British HCI Group Annual Conf. People and Comput.*, Lancaster, UK, 3-7 September 2007.
[28] K. Kautz, "Participatory Design Activities and Agile Software Development," in *Int. Working Conf. IFIP WG*, vol. 318, pp. 303–316, Springer, 2010.
[29] M. Fernández, D. Penzenstadler, B. Kuhrmann, M. Broy, "A Meta Model for Artefact-Orientation: Fundamentals and Lessons Learned in Requirements Engineering," in D. C. Petriu, N. Rouquette, Ø. Haugen, Model Driven Engineering Languages and Systems, MODELS 2010, *Lecture Notes in Comput. Sci.*, vol. 6395, Springer, Berlin.
[30] M. Maguire, "Using Human Factors Standards to Support User Experience and Agile Design," in *Int. Conf. UAHCI 2013*, Las Vegas, USA, 2013.
[31] J. M. Rivero, J. Grigera, G. Rossi, E. Robles Luna, F. Montero and M. Gaedke, "Mockup-Driven Development: Providing agile support for Model-Driven Web Engineering," *Inform. and Softw. Technol.*, vol. 56, no. 6, pp. 670-687, 2014.
[32] E. M. Schön, J. Thomaschewski and M. J. Escalona, "Identifying Agile Requirements Engineering Patterns in Industry by Means of Empirical Research," in *22nd Europ. Conf. Patt. Langua. of Progr.*, 2017.
[33] E. M. Schön, T. Jörg and M. J. Escalona, "Agile Requirements Engineering: A Systematic Literature Review," *Comput. Standar. & Interf.*, vol. 49, pp. 79-91, 2017.
[34] E. Kheirkhah and A. Deraman, "Requirements Engineering in End-User Computing: A review," in *2008 Int. Symp. on Inform. Technol.*, Kuala Lumpur, Malaysia, pp. 1-8, 2008.
[35] A. Gupta, G. Poels and P. Bera, "Using Conceptual Models in Agile Software Development: A Possible Solution to Requirements Engineering Challenges in Agile Projects," *IEEE Access*, vol. 10, pp. 119745-119766, 2022.
[36] A. Hamzah, M. Omar and R. Romli. "The State of The Art of Agile Kanban Method: Challenges and Opportunities," *Independ. J. Manage. & Product.*, vol. 12, no. 8, pp. 2535-2550, 2021.