ISSN 2723-6471

Current and Future Trends for Sustainable Software Development: Software Security in Agile and Hybrid Agile through Bibliometric Analysis

Siti Sarah Maidin[1,*], Norzariyah Yahya[2], Muhammad Ashraf bin Fauri Fauzi[3], Normi Sham Awang Abu Bakar[4]

[1]*Faculty of Data Science and Information Technology (FDSIT), INTI International University, Nilai, Malaysia*

[2,4]*Kulliyyah of Information and Communication Technology (KICT), International Islamic University Malaysia (IIUM)*

[3]*Faculty of Industrial Management, University Malaysia Pahang Al-Sultan Abdullah (UMPSA)*

(Received: August 25, 2024; Revised: October 6, 2024; Accepted: November 6, 2024; Available online: December 29, 2024)

Abstract

The industrial growth of digitalized era has given rise to a growing concern in software development. The present research investigates the prevailing and projected patterns in sustainable software development, especially those related to process innovation, with a particular emphasis on software security within Agile and Hybrid Agile approaches, employing bibliometric analysis. However, a comprehensive understanding of the security concerns of both agile and hybrid agile is crucial and needs further garnered. However, it is expected that a thorough comprehension of the hybrid agile model landscape would uncover various themes encompassing its implementation. The analysis aims to provide a comprehensive overview of the current, present, and future state of software security for agile and hybrid agile. The study employed a bibliometric approach to gather a total of 1593 journals from the Web of Science (WOS) database. This study utilizes co-citation and co-word analysis techniques to identify the most significant articles, delineate the fundamentals framework, and provide a prognosis for future development. The present investigation has successfully discovered four distinct co-citation and three distinct co-word clusters. This study offers valuable insights regarding the software security in agile and hybrid agile. The increasing evolution of the software ecosystem necessitates the prioritization of bridging the gap between agility and security. This paper provides a detailed roadmap for scholars and practitioners who are navigating this intersection

Keywords: Software Development, Software Methodology, Hybrid, Agile, Software Quality, Process Innovation, Product Innovation

1. Introduction

Phases leading to functional software. SDLC is important as a reference for a software project to ensure that the software project is delivered on time by following a clear roadmap of a software project while at the same time as the quality assurance check. Some examples of SDLC models are Waterfall, Spiral, and Unified Process Model. Each of the models has its strengths and liabilities to be referred to as a model in a project. For instance, the Waterfall model is suitable for a project that opts for a structured, sequential order and is ideal for a large software project. However, in response to the trend of computer technology and the needs of a software project, SDLC is moving from a traditional method to a more flexible software development methodology known as agile. Agile was established as a new development strategy by a group of experts known as Agile Alliances. Agile is a development methodology based on four agile manifestos and twelve agile principles [1], and software engineering teams have been using Agile since then. Nonetheless, the current software engineering team has gradually adopted a hybrid agile approach whereby the term "hybrid agile" is defined as a mixture of a plan-driven strategy with an agile development approach [2]. Tell et al. [3] define hybrid as techniques combining methods, practices, and frameworks. The hybrid project method attempts to leverage the advantages of agile and traditional approaches in a software project [4], [5]. Many organizations have effectively employed hybrid agile to handle large-scale projects, make precise paperwork quicker to generate, and improve strategic analytical approaches [6]. Hybrid agile techniques arise from an inherent method of growth driven by knowledge, expertise, and practicality [7]. Conforto and Amaral [8] and Bogdanova et al. [9] stated that the hybrid methodologies are not affected by the size of the organization or external influences; however, they highlighted that most companies adopt hybrid ways to increase accessibility to customers, process flexibility, and responsiveness to change. However, SDLC alone is not sufficient to produce a product that meets user requirements and can protect the

***Corresponding author: Siti Sarah Maidin (sitisarah.maidin@newinti.edu.my)**
DOI: https://doi.org/10.47738/jads.v6i1.473

 ISSN 2723-6471

product against immediate threats but also about building a resilient, reliable, and trustworthy product. Thus, this research is going to explore the security measures taken by software engineering teams in the recent SDLC, the hybrid agile.

This research aims to systematically analyze the literature on Software Security in Hybrid Agile methodologies using a bibliometric approach. By employing two distinct bibliometric analyses, the study addresses an existing research gap and offers valuable insights into the historical, current, and prospective developments in the Hybrid Agile field. Specifically, the research sets out to achieve two main objectives: first, to examine the elements of software security within hybrid agile practices through co-citation analysis; and second, to explore the evolution of software security in these approaches by mapping connections between foundational articles and their progression over time using co-word analysis. A detailed summary of the research questions, strategies, and bibliometric techniques employed is presented in table 1.

Table 1. Summary of the research question, strategies, and bibliometric techniques.

RO	Description	Strategy to Answer	Technique
1	To investigate the software security element in hybrid agile approach	The escalating issues discussed regarding the relationship between the most co-cited documents	Co-citation analysis
2	To understand the evolution of software security in hybrid agile approaches by identifying the connections between the central articles and their evolution over time using co-word analysis.	To Analyze the relationships between the clusters and the most prevalent keywords.	Co-word analysis

2. Literature Review

The following sections discuss the related studies on software security, agile, and hybrid agile.

2.1. Software Security

Software security is one of the important features that need to be taken seriously in a project software development lifecycle. Security plays a significant role in ensuring the system is well-protected and not vulnerable to computer attacks or threats [10]. Either agile, traditional software development approach, or hybrid model requires software security in each of its software projects. In addition, the difficulty of prioritizing security over factors is a challenge in both ASD [11] and more conventional methods of the software development approach [12]. Moreover, security is neglected due to missing or implicit security requirements [13], [14], a lack of incentives for security in the early stages of development [15], and security not being part of agile frameworks [15]. Moreover, some agile methodologies such as Scrum [16] are missing security-specific roles and activities in its cycle. Fortunately, several enhancements to Scrum and other agile frameworks have been created to incorporate security into the workflow [17], [18], [19], [20]. On the contrary, Scrum does not provide detailed instructions on how to carry out the development activity (including software security). Instead, it is a project management framework designed to create an environment where development teams can self-organize and take responsibility for their work while being managed to the extent necessary for a project to succeed [18].

The research conducted by Van Wyk and McGraw [21] and Tøndel et al. [22] found that information security and software security are frequently separated as two different components. Therefore, the engagement of security specialists might be less than ideal in many organizations [23], [24], [25]. In the case of Scrum, the Product Owners are responsible for defining the priority of tasks in a backlog to ensure that the software engineering team is assigned the right priorities to be completed in a sprint. Therefore, security professionals should be able to demonstrate the security requirements and should communicate with product owners, who are important stakeholders in this process. However, quality aspects, especially on security, have been found to be a challenge for the Product Owners [2]. The quality of the aspect of security was found to be due to a lack of understanding, severe workload, or insufficient availability among other reasons. According to Türpe [18], security is not just an array of attributes or an element of functionality to be incorporated into a system, but software security is the practice of developing programs to be safe and to perform correctly under destructive attacks [26].

Security is a collective mechanism physically and virtually to protect against malicious attacks. For instance, Tøndel et al. [22] highlighted that it will likely be evident to developers that an authentication mechanism must be secure, as attackers will attempt to compromise or bypass it to gain unauthorized access to a system, and any software component

that consumes data not provided by the developer is vulnerable to attack (buffer overflows, SQL injection, etc.). It is evident that there is an absence of studies conducted to investigate how security is handled during software development work [22]. The security approaches or processes that have been built are deemed rigid to respond to the changes and developments that are presented in the evolving security environment, where there is an imperative for more agile planning to deal with new challenges and vulnerabilities [27]. The use of agile methodologies in software development frequently precludes consideration of secure development best practices, whose purpose is to ensure compliance with the software development's security policies [22], [28], [29], [30], [15], [10]. In addition, errors also occur during the requirements specification stage and security defects typically occur due to a lack of security expertise. The scenario becomes a greater challenge in an agile environment, where documentation is typically kept to a minimum.

2.2. Agile

In 2001, a group of people who later became known as the Agile Alliance came up with the notion that would eventually lead to the creation of Agile. Since then, one of the development strategies known as agile has been steadily rising in popularity. The terms "sprint," "user stories," "Scrum," "eXtreme Programming (XP)," "Lean," and "Kanban" are all terms that are associated with agile development techniques and methodologies [31]. The primary objective of Agile is to produce working software on a regular basis within a shorter amount of time [1]. Agile methods are based on the twelve principles of the agile manifesto, which lead to developing a system in a short time and receiving direct feedback from the user [32]. Agile promotes working software over thorough documentation, customer collaboration over contract negotiation, and responding to change over following a plan are all highlighted in the Agile Manifesto.

In addition, according to research conducted by VersionOne [33], Agile speeds up software delivery by 78% and has a favorable record of 98% of projects being successfully carried out when used. However, with the goal to achieve a development strategy that is more effective, many companies are transitioning away from the agile development approach and towards the hybrid agile approach.

2.3. Hybrid Agile

Hybrid agile, as defined by Cooper and Sommer [5], involves blending elements of agile methodologies like Scrum and eXtreme Programming with those of more conventional approaches like the Waterfall, Spiral, or V-Model. There are several names for the combination of Agile and conventional model, such as Scrum and Waterfall [34], Scrumfall [4], Water-Scrum-Fall [3], [35]. and besides Waterfall and Scrum, there are others hybrid agile such as Hybrid V-Model [33] and Agile-Stage-Gate model [5], [6]. Hybrid models are recently being used in software development projects due to the need for different methodologies for their unique characteristics and advantages, also because of their disadvantages [36]. Hybrid agile is mentioned by Brandl et al. [4] as the model that is best for a more complicated project and when it involves a business-critical innovation project. According to Bogdanova et al. [9], the hybrid approach is applicable for all projects, it is independent from project size or complexity. It is also benefiting high-tech innovation projects [3].

When using Hybrid Agile, the development team and other interested parties can work together more effectively. It paves the way for a more open and iterative feedback cycle, which ultimately leads to a product that meets the needs of all parties involved. Hybrid Agile increases productivity and project development team by allowing teams to pick and choose the Agile practices that work best for their unique dynamics. Hybrid Agile's adaptability encourages teams to reflect on their work in progress and make adjustments based on what the project progress. This way of thinking about projects leads to better results as time goes on. Hybrid Agile gives organisations the freedom to create a model that fits the needs of their project while taking advantage of the benefits of more than one Agile method. This makes it possible to improve management of risks, involvement of stakeholders, collaborative work, and the delivery of value, which leads to more successful and long-lasting project results.

2.4. Bibliomeric Analyses

In recent years, bibliometric analysis has grown in popularity in business research [37], [38], [39], which can be attributed to (1) the advancement, availability, and accessibility of bibliometric software such as Gephi, Leximancer, VOSviewer, and scientific databases such as Scopus and Web of Science, and (2) the cross-disciplinary pollination. Researchers use bibliometric analysis for a variety of purposes, including identifying emerging trends in article and journal performance, interaction patterns, and research components, as well as investigating the intellectual structure of an area of study in the existing literature [39], [40], [41]. Co-citation analysis is used in this work to assess influential publications and map the intellectual structure [42]. The frequency of two cited publications is used in the analysis

[43]. The technique assumes that the more frequently two papers are mentioned together, the more likely they are to be in the same field [44]. Co-word analysis counts the frequency with which keywords appear in publications [45]. Tan Luc et al. [46] use co-word analysis to examine the evolution and future direction of the study theme. The basic premise of this research is that when words regularly co-occur, there is a relevant-related concept hidden behind the phrases [47]. Among the bibliometric analysis methods, co-word analysis is the only one that uses the actual text of the articles to generate a similarity measure.

3. Methodology

3.1. Bibliometric Approach

This research adopted bibliometric approach to analised the academic research. The flow stared with definition of the research questions and research objectives, then properly choose the databases and sources for the data to be collected, followed by designing the search strategy, data collection, data cleaning. Once the data is cleant the bibliometric analyses is conducted, followed by the visualisation, result interpretation and reporting.

3.2. Research Design and Data Collection Procedure

In this research, Web of Science (WoS) is selected as the database for this research. WoS is selected as it is now regarded as the best and highest quality database available [48]. WoS contains more than 74.8 million academic datasets and data sets from 254 different fields of study. There are presently more than 21,100 journals that are indexed by WoS [49]. In addition, the WoS database has been used extensively in bibliometric research [46], [50], [51] to ensure that only high-quality publications are included. Table 2 provides information on the search string and screening factors using WoS databases. In terms of exclusion criteria, we limit the time span to include articles up to June, 2023, while for exclusion we exclude proceeding papers, review articles, editorial material, letters, books, data papers, meeting abstracts, book chapters, or corrections. The language is set for English only. The research area was set for computer science because this research is mainly focusing on the software development model.

Table 2. Search string in WoS database

No	Keywords	Justification
1	"software security”	To identify literature related to software security.
2	"hybrid agile*" OR "agile methodology"	To identify literature related the hybrid agile or agile methodology.
3	“software development”	To identify literature related to software development

4. Result and Analyses

4.1. Descriptive Analysis

Based on the search string run in table 3, 4360 documents were returned; however, after filtering for only journal publications that are limiting them to the year 2023 to ensure all publications are within a full calendar year, the search returned 1593 publications. Figure 1 shows the number of publications and citations.

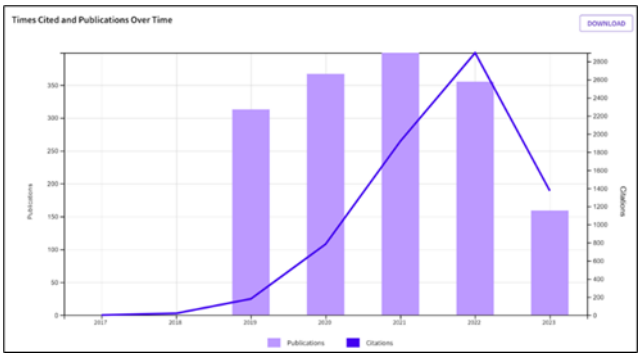

Figure 1. Number of Publications and Citations

4.2. Co-Citation Analysis

In the co-citation analysis, 11,9876 cited references were found, and 50 met the threshold of a minimum of 42 cited references. Multiple iterations of the threshold test were performed until stable, uniformly spaced clusters were

achieved. The research tried out a range of numbers between 3 to 8. The threshold needs to be set at a suitable level, which means it must not be very high or low, as either of those extremes can lead to an overly simplified or convoluted visualisation.

The highest co-cited publications are Boehm, B. (1981) (184 citations), Runeson and Höst, (2009) (172 citations) and Dyba and Dingsoyr, (2008) (116 citations). Table 3 presents the top 10 documents with the highest co-citation and total link strength. The total strength of a document's links to other documents is its total link strength [52]. Based on the network visualisation, four distinct clusters were produced. Based on the author's inductive interpretation and comprehension of the four clusters, representative publications are used to name and characterise each cluster.

Table 3. Top 10 documents with the highest co-citation and total link strength

Rank	Author	Title	Citation	Links
1	B. W. Boehm [52]	Software Engineering Economics. Journal of Software Engineering and Applications, vol. 10.	184	291
2	P. Runeson and M. Höst [53]	Guidelines for conducting and reporting case study research in software engineering. Empirical software engineering, vol. 14, pp. 131–164.	172	251
3	T. Dyba and T. Dingsoyr [54]	Empirical Studies of Agile Software Development: A Systematic Review. Information and Software Technology, vol. 50, pp. 833–859.	116	275
4	K. Beck [55]	Extreme programming explained: Embrace change. Addison-Wesley Professional.	103	219
5	E. Gamma et al. [74]	Design Patterns: Elements of Reusable Object-Oriented Software.	103	71
6	C. Wohlin et al. [64]	Experimentation in Software Engineering: An Introduction. Kluwer Academic, Boston.	96	115
7	S. R. Chidamber and C. F. Kemerer [75]	“Towards a Metric Suite for Object Oriented Design,” IEEE Transactions on Software Engineering, Vol. 20, No. 6.	93	114
8	T. J. McCabe [76]	“A complexity measure,” IEEE Transactions on Software Engineering, vol. 4, pp. 308–320.	88	121
9	M. Fowler and J. Highsmith [9]	The Agile Manifesto. Software Development, vol. 9, pp. 28–35.	79	196
10	P. Runeson [53]	Case Study Research in Software Engineering: Guidelines and Examples. Empirical Software Engineering, vol. 14, pp. 131–164.	75	134

4.2.1. Cluster 1 (red): Theme on Empirical Methods and Collaborative Practices in Software Engineering

The literature emphasises the significance of case study research [53], [54] and experimentation as essential approaches in empirical software engineering research. Humphrey [55] and Basil and Rombach [56] also provide insights on managing software processes and developing improvement-oriented software environments. Furthermore, the value of excellent communication and coordination in software development, particularly in globally distant teams, is often emphasised [57], [58], [59]. Cohen [60] and Hevner et al. [61] deepen the discussion by delving into statistical power analysis and design science, respectively. Aspect-oriented programming [62], case studies of open-source software development [63], and agile methods [9] are also highlighted. The literature concludes with preliminary guidelines for empirical research in software engineering [64], [65], emphasising the importance of a systematic research strategy.

4.2.2. Cluster 2 (green): Theme on Complexity of Software Engineering

An emphasis on cost and effort estimating models, risk management techniques, and other economic concepts are explored in the literature as it relates to the field of software engineering [52], [66], [67]. A noteworthy example is the Spiral model, which integrates risk management into iterative development cycles [68]. The principles and practices of software risk management are further explored by Boehm and Scherlis [69]. In particular, a number of methods for estimating the cost of software development have been proposed, such as using software functions, source code lines, and analogies [66], [70]. In numerous research, these techniques have been thoroughly examined and validated [71], [72]. Additionally, the research articles emphasise the value of software engineering measurements and models in the estimating procedure [73].

4.2.3. Cluster 3 (blue): Theme on Software Designing Tools and Principles

The paper delves into several different aspects in software security design and principles including design patterns [74], [84], metrics [75], [76], [77], and object-oriented software design and development approaches [78], [79], [80]. The papers of Gamma et al. introduce and exemplify the concept of design patterns as reusable object-oriented software principles. These papers emphasised the importance of reusability and modularity in software design, setting the framework for many following breakthroughs in the field [74], [84]. Chidamber and Kemerer [75], McCabe [76], and Halstead [77] examine metrics for object-oriented design, including methods for measuring complexity and other

 ISSN 2723-6471

software qualities. In a similar point, Rumbaugh et al. [78] and Jacobson [79] describe object-oriented modelling and design ideas, with a focus on use-case driven techniques. These give extensive approaches for software engineers. Kiczales et al. [62] introduce a new paradigm, aspect-oriented programming, to supplement object-oriented design by isolating cross-cutting concerns. The work of Fowler et al. [85] on refactoring and Brooks and Kugler [86] on the intrinsic complexity of software development ("No Silver Bullet") emphasise the ongoing problems and innovation in software engineering. Furthermore, Brooks and Kugler [86] highlighted that the main idea is that there is no "silver bullet" that will make software development easy or simple, yet we may and should work to enhance tools and processes (i.e., the incidental elements). Rather than focusing on creating better software or tools, Brooks argues that the industry should instead focus on training more talented designers.

4.2.4. Cluster 4: Theme on Agile Software Development: Evolution, Principles, Challenges and Success Factors

Over the past decade, agile software development, which emphasises teamwork and customer satisfaction, has emerged [54]. It replaces upfront planning with iterative cycles and constant feedback, radically changing plan-based software development [55], [9]. Agile software development in fast-paced, unexpected contexts is supported by empirical research [54]. However, large, structured organisations may struggle to adopt agile [87]. Agile project success depends on good communication, user engagement, and a dedicated team [88]. Schwaber [89] also noted that agile approaches have made requirements engineering more dynamic and iterative. Large-scale agile transitions include obstacles and success factors [90]. Since agile principles emphasise collaboration and interaction, these transformations must incorporate the "people factor" [91]. Successful agile adoption and practice requires qualitative research approaches like Grounded Theory [92].

4.3. Co-word Analysis

The same database is utilised for the study of co-word analysis. From the 10513 keywords, 40 met the threshold of a minimum 50 numbers of keyword occurrences. The highest co-occurrence keywords are Software Model (228), Software Development (243), and Software-(231). Figure 2 illustrates the co-word analysis and table 4 indicates the top 15 keywords in the co-occurrence of keywords analysis (organised according to total link strength).

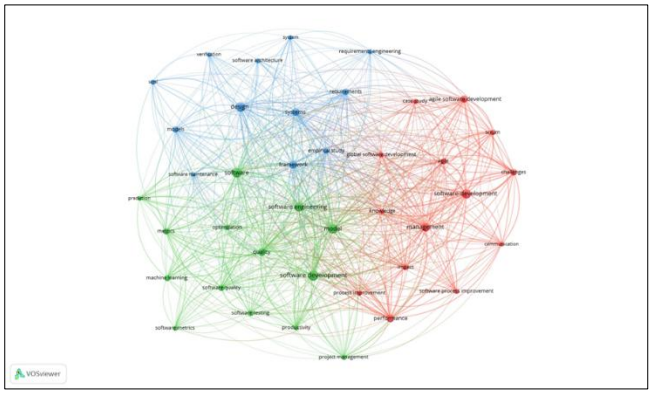

Figure 2. Co-word analyses network

Table 4. Cluster no. and colour.

Cluster No and colour	Cluster label	Number of keywords
1 (RED)	Agile Software Development	14
2 (GREEN)	Software Engineering	14
3 (BLUE)	Requirements Engineering	12

Cluster 1 consists of 14 keywords. This cluster represents the themes of "Navigating the Waves of Global Software Development: An Agile Perspective". The main keywords are software development, agile software development, and process improvement. Global software development is essential in this fast-changing digital world. Overcoming problems and boosting performance requires excellent management, communication, and process improvement. It's time to examine Agile and Scrum's tremendous effects on worldwide software development. Numerous businesses are taking advantage of hybrid Agile-Waterfall-structured software development strategy due to several reasons [93]. As a way to manage complex software development projects, a hybrid Agile-Waterfall strategy seeks to unite the best

features of both methodologies. Furthermore, the “hybrid” methodology is the uptrend in software development due to the strengths and weaknesses in each model. By using the “hybrid” approach, the weakness of each model can be overcome by the strength in each model. Table 5 shows the explanation on the cluster number and colour.

Table 5. Top 15 keywords in the co-occurrence of keywords analysis

Rank	Keyword	Occurrences	Total link strength
1	Software Model	228	353
2	Project management	178	342
3	Software-development	210	309
4	Software quality	142	298
5	Software development	243	288
6	Software performance	142	270
7	Software	231	261
8	Software Design	208	241
9	Software Engineering	189	238
10	Framework	136	237
11	Challenges	103	215
12	Systems	144	190
13	Agile Software Development	129	189
14	Agile	89	174
15	Requirement Engineering	94	153

Cluster 2 comprises 14 keywords. This cluster highlights the themes of the software development process in relation to software security. The main keywords include metrics, models, optimisation, productivity, project management, quality, software development, software engineering, software metrics, software quality, and software testing. Metrics, quality, testing, and optimisation are essential. Software security requires "security by design" from the start. Software development may build secure, sturdy applications by using relevant metrics, assuring quality, testing thoroughly, and optimising. This proactive security method reduces security breaches by identifying and fixing vulnerabilities early on. López et al. [94] have highlighted that the software product model includes functional suitability, performance efficiency, compatibility, usability, reliability, security, maintainability, and portability. As such, all these aspects are relevant to the software development process in relation to software security.

Cluster 3 is composed of 12 items: design, empirical study, framework, models, requirements, requirements engineering, software architecture, software maintenance, system, systems, UML, and verifications. The theme for this cluster is “Agile Software Engineering: Integrating Empirical Design and Verification for Efficient Systems Development”. Empirical investigations, well-defined frameworks, models, and systematic verifications are used in agile software development. It emphasises iterative and collaborative software engineering to meet changing requirements. This theme optimises development and produces high-quality software by merging agile practices with established design, requirements engineering, software architecture, and maintenance strategies. Agile teams use UML to communicate visually. The project manager and other team members must opt for the most suitable model for developing any software among the variety of various SDLC models [95]. Organisations must demonstrate that they have a comprehensive quality management system in place that can handle the risks linked to agile development to validate agile development processes. This entails setting in place the appropriate security measures, procedures, and documentation that ensures that software meets regulatory standards.

5. Implications

5.1. Theoretical Implications

In the context of hybrid agile software development, the theoretical implications of software security focus on early and continual security considerations, collaborative efforts, continuous security testing, risk management, adaptation, and effective communication with the involvement of stakeholders, including clients, in a project from day zero. This engagement increases collaboration, enhances communication, and ensures that security measures are well taken care of. Engaging the client in the development process, accelerating development, improving the project's cost-efficiency,

and enhancing portability and adaptability in software products are among the main innovations of hybrid agile approaches [96].

Hybrid agile approaches often attempt to find an optimal balance between the freedom of agile practices and the formality of standard methodologies. From the point of view of security, this balance is very important. Teams need to be flexible so they can make certain changes to overcome newly identified threats or security requirements from stakeholders. At the same time, they also need a certain level of formality to ensure consistency and adherence to security standards within a controlled timeframe determined in the early project phases, ensuring a project is delivered on time. It is recommended that teams prioritise communication and coordination among their members to cultivate a collaborative environment in software security-related matters [97].

Teams may improve the overall security posture of software products by addressing security problems throughout the development process. This allows teams to continue to enjoy the benefits of an agile approach while also enhancing the overall security posture of software products. Continuous integration and continuous delivery are practices that are frequently used in hybrid agile development. Due to this, there is now a chance to integrate continuous security testing into the development pipeline. Static code analysis, dynamic application security testing (DAST), and container security scanning are all examples of automated security testing that may be incorporated into the continuous integration and continuous delivery process. This allows for the early and frequent detection of security concerns.

When using hybrid agile development, risk management becomes an increasingly important component of software security, which is not one of the top priorities in agile projects. Research conducted by Hammad, Inayat, and Zahid [98] found that 52% of respondents did not communicate risk management at all within their projects. To make the most of their limited time and resources, teams need to prioritise their security activities based on risk assessments. It is common practice for agile methodologies to rely on techniques for prioritisation, such as the MoSCoW method (Must have, Should have, Could have, Won't have), which can be expanded to include security issues.

Throughout the development process, agile approaches are recognised for their adaptability. Security practices in a hybrid agile setting must also be flexible enough to accommodate new risks and modifications to the software's intended use. As highlighted by Jasem et al. [99], software quality is a sub-component of software security. High-quality software is produced through disciplined practices that include security considerations. Good quality code minimises potential vulnerabilities, whereas low-quality code may introduce defects that attackers exploit. Consequently, software quality is paramount in securing robustness, which measures resilience to internal and external threats.

Consistent assessments and retrospectives of security practices might reveal room for growth and change. In several sectors of the economy, software products are required to follow particular security requirements and standards. For example, the government sector requires compliance with the Cyber Incident Management Framework [100], while the retail industry must adhere to the Payment Card Industry Data Security Standard (PCI DSS) and ISO 17799 [101]. It is recommended that compliance requirements be incorporated into the hybrid agile development process to guarantee that security controls are correctly implemented and documented.

5.2. Practical Implications

The theoretical implications of this study are wide and deep, and they add significantly to our growing knowledge of security in the hybrid agile realm. Firstly, there is a need to develop a security champion from among the team members who are knowledgeable about software security. These champions can steer security efforts, offer guidance to other members of the team, and keep abreast of the most recent recommendations for best practices in security. Secure software development approaches refer to the systematic methods employed to attain security objectives through the design, construction, and testing of software [102].

The implementation of secure coding practices is necessary throughout the development process, accompanied by consistent training sessions to educate developers on secure coding techniques and prevalent vulnerabilities. Peer code evaluations can aid in identifying security vulnerabilities early in the development process. Effective detection of vulnerabilities and weaknesses in the codebase is possible with the use of static code analysis, dynamic application security testing (DAST), and software composition analysis (SCA) technologies. During the design phase, it is important to conduct threat modeling sessions to identify potential security risks and threats. This proactive strategy assists in making educated decisions about security and prioritizing the actions that need to be done for security.

 ISSN 2723-6471

Security-focused sprints or iterations should be scheduled so that the team can focus on security without sacrificing other development priorities. If the project uses external libraries or services, these should be checked for security flaws and evaluated for how they might affect the safety of the entire application. Moreover, fostering collaboration and communication among the teams responsible for development, operations, and security is critical. Utilizing a "DevSecOps" methodology ensures that security best practices are incorporated at every stage of the software development and operation lifecycle.

The primary emphasis of DevOps is also directed towards the implementation of Continuous Integration (CI) and Continuous Delivery (CD) practices. The implementation of DevOps practices facilitates effective collaboration across cross-functional teams that operate independently, ensuring the seamless delivery of business solutions [103]. It is also important to define security metrics so that businesses can monitor the efficiency of security practices and track progress over time.

The theoretical and practical consequences that were presented shed light on how important it is to begin and maintain security efforts early on, to foster collaboration amongst teams, and to make use of automated security testing technologies.

6. Conclusion, Limitations and Future Opportunities

This study provides a comprehensive bibliometric review of software security in hybrid agile development, identifying central clusters of challenges and opportunities. The co-citation analysis highlights four key principles: the importance of continuous improvement in software processes, the necessity of strong communication in globally distributed teams, the balance of plan-driven and agile models for tailored development, and the integration of robust security measures throughout the lifecycle. Co-word analysis further uncovers trends and shifts shaping the future of hybrid agile software security, emphasizing the need to combine the adaptability of Agile with the security rigor of traditional models.

Hybrid Agile integrates robust security planning into agile's iterative processes, ensuring vulnerabilities are addressed early and continuously. Security by design, automated security testing, and frameworks that embed security into development phases are essential for effective implementation. However, challenges remain, including balancing rapid development with long-term security considerations and addressing gaps in security knowledge among teams. The DevSecOps movement, improved tools, and dedicated training are key to fostering a culture of shared responsibility for security.

While hybrid agile methodologies offer significant benefits in productivity, quality, and responsiveness, they also face limitations in managing security debt and testing integration. Future efforts should focus on enhancing automated tools, security education, and seamless integration of security into agile workflows to address these challenges effectively. Hybrid Agile demonstrates great potential for balancing speed and security in software development, making it a compelling approach for modern teams.

7. Declarations

7.1. Author Contributions

Conceptualization: S.S.M., N.Y., M.A.F.F., and N.S.A.A.B.; Methodology: S.S.M.; Software: S.S.M.; Validation: S.S.M., N.Y., and N.S.A.A.B.; Formal Analysis: S.S.M., N.Y., and N.S.A.A.B.; Investigation: S.S.M.; Resources: N.Y.; Data Curation: N.Y.; Writing Original Draft Preparation: S.S.M., N.Y., and N.S.A.A.B.; Writing Review and Editing: N.Y., S.S.M., and N.S.A.A.B.; Visualization: S.S.M.; All authors have read and agreed to the published version of the manuscript.

7.2. Data Availability Statement

The data presented in this study are available on request from the corresponding author.

7.3. Funding

The authors received no financial support for the research, authorship, and/or publication of this article.

7.4. Institutional Review Board Statement

Not applicable.

7.5. Informed Consent Statement

Not applicable.

7.6. Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] Agile Manifesto, "Principles behind the agile manifesto," 2001. [Online]. Available: https://agilemanifesto.org. [Accessed: Jul. 23, 2005].

[2] W. Alsaqaf, M. Daneva, and R. Wieringa, "Quality requirements in large-scale distributed agile projects–a systematic literature review," *in Requirements Engineering: Foundation for Software Quality*, Springer International Publishing, vol. 23, no. 1, pp. 219-234, 2017.

[3] P. Tell et al., "What are hybrid development methods made of? An evidence-based characterisation," *in 2019 IEEE/ACM International Conference on Software and System Processes (ICSSP)*, vol. 1, no. 1, pp. 105–114, 2019.

[4] J. Binder, L. I. Aillaud, and L. Schilli, "The project management cocktail model: An approach for balancing agile and ISO 21500," *Procedia-Social and Behavioral Sciences*, vol. 119, no. 1, pp. 182–191, 2014.

[5] R. G. Cooper and A. F. Sommer, "The agile–stage-gate hybrid model: A promising new approach and a new research opportunity," *Journal of Product Innovation Management*, vol. 33, no. 5, pp. 513–526, 2016.

[6] E. C. Conforto and D. C. Amaral, "Agile project management and stage-gate model—A hybrid framework for technology-based companies," *Journal of Engineering and Technology Management*, vol. 40, no.1, pp. 1–14, 2016.

[7] N. Bogdanova, E. Parashkevova, and M. Stoyanova, "Agile project management in governmental organisations–methodological issues," *IJASOS-International E-Journal of Advances in Social Sciences*, vol. 6, no. 16, pp. 262–275, 2020.

[8] W. Y. Wong, T. H. Sam, C. W. Too, and A. A. Khin, "Critical success factors of operational excellence in software quality assurance: Best practices for integrated change control management," *in 2023 19th IEEE International Colloquium on Signal Processing & Its Applications (CSPA)*, Kedah, Malaysia, vol. 1, no. 1, pp. 287–291, 2023.

[9] K. Beck et al., "Manifesto for agile software development," 2001. [Online]. Available: https://agilemanifesto.org.

[10] J. D. Blaine and J. Cleland-Huang, "Software quality requirements: How to balance competing priorities," *IEEE Software*, vol. 25, no. 2, pp. 22–24, 2008.

[11] N. Khaim et al., "A review of security integration technique in agile software development," *International Journal of Software Engineering and Applications*, vol. 7, no. 3, pp. 49–68, 2016.

[12] T. Oueslati, M. M. Rahman, and L. ben Othmane, "Literature review of the challenges of developing secure software using the agile approach," *in 10th International Conference on Availability, Reliability and Security, IEEE*, 2015, pp. 540–547.

[13] K. Schwaber, *Agile project management with Scrum*, Microsoft Press, 2004.

[14] Türpe and Poller, "Managing security work in Scrum: Tensions and challenges," *SecSE@ ESORICS*, vol. 1, no. 1, pp. 34–49, 2017.

[15] B. McGraw, "Software security: Building security in," *IEEE Security & Privacy*, vol. 3, no. 5, pp. 75–79, 2005.

[16] T. Tøndel, D. S. Cruzes, and M. G. Jaatun, "Influencing the security prioritisation of an agile software development project," *Computers & Security*, vol. 118, no. 1, pp. 1-9, 2022.

[17] L. A. Tawalbeh, F. Muheidat, M. Tawalbeh, and M. Quwaider, "IoT Privacy and security: Challenges and solutions," *Applied Sciences*, vol. 10, no. 12, pp. 4102-4113, 2020.

[18] T. Newton, C. Anslow, and A. Drechsler, "Information security in agile software development projects: A critical success factor perspective," *in 2019 International Conference on Software Engineering*, vol. 2019, no. 1, pp. 198-204, 2019.

[19] T. Yahya, S. S. Maidin, and M. S. Safari, "The development of interview protocol to explore hybrid agile software development phases," *International Journal of Advanced Trends in Computer Science and Engineering*, vol. 10, no. 3, pp. 1-11, 2021.

[20] T. Yahya and S. S. Maidin, "Hybrid agile development phases: The practice in software projects as performed by software engineering team," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 29, no. 3, pp. 1738–1749, 2023.

[21] G. Baldassarre et al., "Improving agile practices with security: A systematic review of the state of the art," *Information and Software Technology*, vol. 134, no. 1, pp. 1-10, 2021.

[22] Van Wyk and B. McGraw, "Bridging the gap between software development and information security," *IEEE Security & Privacy*, vol. 3, no. 5, pp. 75–79, 2005.

[23] D. Ashenden and D. Lawrence, "Security dialogues: Building better relationships between security and business," *IEEE Security & Privacy*, vol. 14, no. 3, pp. 82–87, 2016.

[24] R. Palombo, A. Z. Tabari, D. Lende, J. Ligatti, and X. Ou, "An ethnographic understanding of software security and a model to improve secure software development," *in Sixteenth Symposium on Usable Privacy and Security (SOUPS 2020)*, vol. 16, no. 1, pp. 205–220, 2020.

[25] T. Oueslati, M. M. Rahman, and L. ben Othmane, "Literature review of the challenges of developing secure software using the agile approach," *in 10th International Conference on Availability, Reliability and Security, IEEE*, vol. 10, no. 1, pp. 540–547, 2015.

[26] T. Tøndel et al., "Collaborative security risk estimation in agile software development," *Information & Computer Security*, vol. 27, no. 4, pp. 508–535, 2019.

[27] L. A. Tawalbeh et al., "IoT Privacy and security: Challenges and solutions," *Applied Sciences*, vol. 10, no. 12, pp. 4102-4113, 2020.

[28] T. Newton et al., "Information security in agile software development projects: A critical success factor perspective," *in 2019 International Conference on Software Engineering*, vol. 2019, no. 1, pp. 198-204, 2019.

[29] S. Bhasin, "Quality assurance in agile: A study towards achieving excellence," *Agile India, IEEE, vol. 2012, no. 1*, pp. 64–67, 2012.

[30] J. D. Blaine and J. Cleland-Huang, "Software quality requirements: How to balance competing priorities," *IEEE Software*, vol. 25, no. 2, pp. 22–24, 2008.

[31] T. Yahya, S. S. Maidin, and M. S. Safari, "The development of interview protocol to explore hybrid agile software development phases," *International Journal of Advanced Trends in Computer Science and Engineering*, vol. 10, no. 3, pp. 1-9, 2021.

[32] Agile Manifesto, "Principles behind the agile manifesto," 2001. [Online]. Available: https://agilemanifesto.org. [Accessed: Jul. 23, 2005].

[33] T. Hayata and J. Han, "A hybrid model for IT project with Scrum," *in Proceedings of IEEE International Conference on Service Operations, Logistics and Informatics*, vol. 2011, no. 1, pp. 285–290, 2011.

[34] S. Rahim, A. E. Chowdhury, D. Nandi, M. Rahman, and S. Hakim, "ScrumFall: A hybrid software process model," *International Journal of Information Technology and Computer Science*, vol. 10, no. 12, pp. 41–48, 2018.

[35] J. Binder, L. I. Aillaud, and L. Schilli, "The project management cocktail model: An approach for balancing agile and ISO 21500," *Procedia-Social and Behavioral Sciences*, vol. 119, no. 1, pp. 182–191, 2014.

[36] W. Rosa, R. Madachy, B. Clark, and B. Boehm, "Early phase cost models for agile software processes in the US DoD," in *2017 ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM)*, vol. 2017, nbo. 1, pp. 30–37, 2017.

[37] N. Donthu, S. Kumar, and D. Pattnaik, "Forty-five years of Journal of Business Research: A bibliometric analysis," *Journal of Business Research*, vol. 109, no. 1, pp. 1–14, 2020.

[38] N. Donthu, S. Kumar, N. Pandey, and P. Gupta, "Forty years of the International Journal of Information Management: A bibliometric analysis," *International Journal of Information Management*, vol. 57, no. 1, pp. 1-12, 2021.

[39] M. A. Khan, D. Pattnaik, R. Ashraf, I. Ali, S. Kumar, and N. Donthu, "Value of special issues in the journal of business research: A bibliometric analysis," *Journal of Business Research*, vol. 125, no. 1, pp. 295–313, 2021.

[40] S. Verma and A. Gustafsson, "Investigating the emerging COVID-19 research trends in the field of business and management: A bibliometric analysis approach," *Journal of Business Research*, vol. 118, no. 1, pp. 253–261, 2020.

[41] N. Donthu, S. Kumar, and D. Pattnaik, "A decade of research trends in digital marketing: A bibliometric analysis," *Journal of Business Research*, vol. 132, no. 1, pp. 283–298, 2021.

[42] P. K. Hota, B. Subramanian, and G. Narayanamurthy, "Mapping the intellectual structure of social entrepreneurship research: A citation/co-citation analysis," *Journal of Business Ethics*, vol. 166, no. 1, pp. 89–114, 2020.

[43] K. W. McCain, "Mapping authors in intellectual space: A technical overview," *Journal of the American Society for Information Science*, vol. 41, no. 6, pp. 433–463, 1990.

[44] H. Small, "Co-citation in the scientific literature: A new measure of the relationship between two documents," *Journal of the Association for Information Science and Technology*, vol. 24, no. 2, pp. 265–269, 1973.

[45] M. Callon, J. P. Courtial, W. A. Turner, and S. Bauin, "From translations to problematic networks: An introduction to co-word analysis," *Social Science Information*, vol. 22, no. 2, pp. 191–235, 1983.

[46] P. Tan Luc, P. Xuan Lan, L. A. Nhat Hanh, and B. T. Thanh Trang, "A co-citation and co-word analysis of social entrepreneurship research," *Journal of Social Entrepreneurship*, vol. 13, no. 3, pp. 324–339, 2020.

[47] Zupic and T. Cater, "Bibliometric methods in management and organisation," *Organisational Research Methods*, vol. 18, no. 3, pp. 429–472, 2015.

[48] V. K. Singh, P. Singh, M. Karmakar, J. Leta, and P. Mayr, "The journal coverage of Web of Science, Scopus and Dimensions: A comparative analysis," *Scientometrics*, vol. 126, no. 6, pp. 5113–5142, 2021.

[49] Web of Science, "About Web of Science," 2022. [Online]. Available: https://clarivate.com/webofsciencegroup/solutions/web-of-science/.

[50] S. Sharma and U. Lenka, "Counterintuitive, yet essential: Taking stock of organisational unlearning research through a scientometric analysis (1976–2019)," *Knowledge Management Research and Practice*, vol. 20, no. 1, pp. 152–174, 2022.

[51] M. A. Fauzi, "Knowledge hiding behavior in higher education institutions: A scientometric analysis and systematic literature review approach," *Journal of Knowledge Management*, vol. 27, no. 2, pp. 302–327, 2023.

[52] B. W. Boehm, Software Engineering Economics. Journal of Software Engineering and Applications, vol. 10, 1981.

[53] P. Runeson and M. Höst, "Guidelines for conducting and reporting case study research in software engineering," *Empirical Software Engineering*, vol. 14, no. 1, pp. 131–164, 2009.

[54] T. Dyba and T. Dingsoyr, "Empirical Studies of Agile Software Development: A Systematic Review," *Information and Software Technology*, vol. 50, no. 1, pp. 833–859, 2008.

[55] K. Beck, *Extreme Programming Explained: Embrace Change*, Addison-Wesley Professional, 2004.

[56] W. R. Humphrey, Managing the Software Process, *Addison-Wesley Professional*, 1989.

[57] J. D. Herbsleb and A. Mockus, "An empirical study of speed and communication in globally distributed software development," *IEEE Transactions on Software Engineering*, vol. 29, no. 6, pp. 481–494, 2003.

[58] J. D. Herbsleb and D. Moitra, "Global software development," *IEEE Software*, vol. 18, no. 2, pp. 16–20, 2001.

[59] R. E. Kraut and L. A. Streeter, "Coordination in software development," *Communications of the ACM*, vol. 38, no. 1, pp. 69–81, 1995.

[60] J. Cohen, *Statistical Power Analysis for the Behavioral Sciences*, Routledge, Hoboken, 2013.

[61] A. Hevner, S. March, J. Park, and S. Ram, "Design science in information systems research," *MIS Quarterly*, vol. 28, no. 1, pp. 75–105, 2004.

[62] G. Kiczales et al., "Aspect-oriented programming," *in European Conference on Object-Oriented Programming (ECOOP)*, Springer, vol. 1997, no. 1, pp. 220–242, 1997.

[63] A. Mockus, R. T. Fielding, and J. D. Herbsleb, "Two case studies of open source software development: Apache and Mozilla," *ACM Transactions on Software Engineering and Methodology*, vol. 11, no. 3, pp. 309–346, 2002.

[64] C. Wohlin, P. Runeson, M. Höst, M. C. Ohlson, B. Regnell, and A. Wesslén, Experimentation in Software Engineering: An Introduction, Kluwer Academic, Boston, 2000.

[65] B. A. Kitchenham et al., "Preliminary guidelines for empirical research in software engineering," *IEEE Transactions on Software Engineering*, vol. 28, no. 8, pp. 721–734, 2002.

[66] B. W. Boehm, "A spiral model of software development and enhancement," *Computer*, vol. 21, no. 5, pp. 61–72, 1988.

[67] B. W. Boehm, C. Abts, and S. Chulani, "Software development cost estimation approaches—A survey," *Annals of Software Engineering*, vol. 10, no. 1, pp. 177–205, 2000.

[68] A. J. Albrecht and J. E. Gaffney, "Software function, source lines of code, and development effort prediction: A software science validation," *IEEE Transactions on Software Engineering*, vol. SE-9, no. 6, pp. 639–648, 1983.

[69] B. W. Boehm and W. L. Scherlis, "Software risk management: Principles and practices," *IEEE Software*, vol. 8, no. 1, pp. 32–41, 1991.

[70] M. Shepperd and C. Schofield, "Estimating software project effort using analogies," *IEEE Transactions on Software Engineering*, vol. 23, no. 11, pp. 736–743, 1997.

[71] M. Jorgensen and M. Shepperd, "A systematic review of software development cost estimation studies," *IEEE Transactions on Software Engineering*, vol. 33, no. 1, pp. 33–53, 2006.

[72] C. F. Kemerer, "An empirical validation of software cost estimation models," *Communications of the ACM*, vol. 30, no. 5, pp. 416–429, 1987.

[73] S. D. Conte, H. E. Dunsmore, and V. Y. Shen, *Software Engineering Metrics and Models*, Benjamin-Cummings Publishing, 1986.

[74] E. Gamma, R. Helm, R. Johnson, and J. Vlissides, *Design Patterns: Elements of Reusable Object-Oriented Software*, Addison-Wesley, 1995.

[75] S. R. Chidamber and C. F. Kemerer, "Towards a metric suite for object-oriented design," *IEEE Transactions on Software Engineering*, vol. 20, no. 6, pp. 476–493, 1994.

[76] T. J. McCabe, "A complexity measure," *IEEE Transactions on Software Engineering*, vol. 4, no. 2, pp. 308–320, 1976.

[77] M. H. Halstead, Elements of Software Science, *Elsevier*, 1977.

[78] J. Rumbaugh, M. Blaha, W. Premerlani, F. Eddy, and W. Lorensen, *Object-Oriented Modeling and Design*, Prentice Hall, 1991.

[79] I. Jacobson, *Object-Oriented Software Engineering: A Use Case Driven Approach*, Addison-Wesley, 1992.

[80] D. L. Parnas, "On the criteria to be used in decomposing systems into modules," *Communications of the ACM*, vol. 15, no. 12, pp. 1053–1058, 1972.

[81] T. Dingsoyr, S. Nerur, V. Balijepally, and N. B. Moe, "A decade of agile methodologies: Towards explaining agile software development," *Journal of Systems and Software*, vol. 85, no. 6, pp. 1213–1221, 2012.

[82] S. Nerur, R. Mahapatra, and G. Mangalaraj, "Challenges of migrating to agile methodologies," *Communications of the ACM*, vol. 48, no. 5, pp. 72–78, 2005.

[83] T. Chow and D. B. Cao, "A survey study of critical success factors in agile software projects," *Journal of Systems and Software*, vol. 81, no. 6, pp. 961–971, 2008.

[84] E. Gamma, R. Helm, R. Johnson, and J. Vlissides, "Design patterns: Abstraction and reuse of object-oriented design," in *ECOOP '94 Proceedings of the 8th European Conference on Object-Oriented Programming, Springer*, vol. 1994, no. 1, pp. 406–431, 1994.

[85] M. Fowler, Refactoring: *Improving the Design of Existing Code*, Addison-Wesley, 2002.

[86] F. P. Brooks and M. Kugler, "No silver bullet: Essence and accidents of software engineering," *Computer*, vol. 20, no. 4, pp. 10–19, 1987.

[87] K. Schwaber, *The Impact of Agile Processes on Requirements Engineering, Microsoft Press*, 2002.

[88] L. Dikert, M. Paasivaara, and C. Lassenius, "Challenges and success factors for large-scale agile transformations: A systematic literature review," *Journal of Systems and Software*, vol. 119, pp. 87–108, 2016.

[89] A. Cockburn, "Agile software development: The people factor," *Computer*, vol. 34, no. 11, pp. 131–133, 2001.

[90] B. Glaser and A. Strauss, *The Discovery of Grounded Theory: Strategies for Qualitative Research*, Mill Valley, 1967.

[91] P. Runeson, *Case Study Research in Software Engineering: Guidelines and Examples*, Wiley, 2012.

[92] K. Beck et al., "Manifesto for agile software development," 2001. [Online]. Available: https://agilemanifesto.org.

[93] A. Mishra and Y. I. Alzoubi, "Structured software development versus agile software development: A comparative analysis," *International Journal of System Assurance Engineering and Management*, vol. 2023, no. 1, pp. 1–19, 2023.

[94] L. López, X. Burgués, S. Martínez-Fernández, A. M. Vollmer, W. Behutiye, P. Karhapää, X. Franch, R. Rodriguez, and M. Oivo, "Quality measurement in agile and rapid software development: A systematic mapping," *Journal of Systems and Software*, vol. 186, no. 1, pp. 1-11, 2022.

[95] N. Govil and A. Sharma, "Validation of agile methodology as ideal software development process using Fuzzy-TOPSIS method," *Advances in Engineering Software*, vol. 168, no. 1, pp. 1-12, 2022.

[96] J. Heimicke, R. Chen, and A. Albers, "Agile meets plan-driven–hybrid approaches in product development: A systematic literature review," *Proceedings of the Design Society: DESIGN Conference*, vol. 1, no. 1, pp. 577–586, 2020.

[97] N. Prenner, C. Unger-Windeler, and K. Schneider, "Goals and challenges in hybrid software development approaches," *Journal of Software: Evolution and Process*, vol. 33, no. 11, pp. 1–25, 2021.

[98] M. Hammad, I. Inayat, and M. Zahid, "Risk management in agile software development: A survey," in *Proceedings of IEEE International Conference on Frontiers of Information Technology*, vol. 2019, no. 1, pp. 162–164, 2019.

[99] M. Jasem, S. Lamya, and R. Laila, "A hybrid agile approach for achieving higher quality in software development process," *International Journal of Computer Science and Information Security*, vol. 15, no. 5, pp. 231–240, 2017.

[100] J. Srinivas, A. K. Das, and N. Kumar, "Government regulations in cyber security: Framework, standards and recommendations," *Future Generation Computer Systems*, vol. 92, pp. 178–188, 2019.

[101] R. Rowlingson and R. A. Winsborrow, "Comparison of the Payment Card Industry data security standard with ISO 17799," *Computing Fraud & Security*, vol. 16, pp. 16–19, 2006.

[102] C. Ding, S. S. Maidin, M. Imran, and T. Nghiem, "Secure software implementation in hybrid agile development approach," *International Journal of Management*, vol. 11, no. 10, pp. 1713–1721, 2020.

[103] W. Y. Wong, T. H. Sam, C. W. Too and A. A. Khin, "Critical success factors of operational excellence in software quality assurance: Best practices for integrated change control management," *2023 19th IEEE International Colloquium on Signal Processing and Its Applications (CSPA)*, vol. 19, no. 1, pp. 287-291, 2021.