Advancing Secure Software Development: Integrative Approaches Across the SDLC with Novel Threat Modeling Techniques

Parit Arvindbhai Jogani
CIS 6370-001: Computer Data Security
Professor: Eric Ackerman

December 9, 2024

Abstract

Software construction is a rapidly evolving process that entails new and complex security risks consequently, scoped security methods must give way to multi-faceted, dynamic approaches to SDLC construction. Traditional approaches mainly address the bring they leave out initial steps like requirement analysis and architecture and implementation. This results in important issues not being reacted to, therefore amplifying prospective dangers and costs.This paper will present the Secure Lifecycle Integration Framework (SLIF), proposing an inside security approach into the SDLC contexts. SLIF uses state of the art threat modeling, improves the UML profiles withcurity annotations, and incorporates the use of predictive analytics solutions to prevent and address risk factors. The objectives of using SLIF are achieved in reduction of vulnerability by 30 percent, cost by 20 percent, and time to market by 15 percent; the framework is consistent with complex development schemes such as Agile and DevOps.Based on real-world examples, it is established that SLIF works and indeed offers the benefits aforementioned. These implications of the presented research suggest SLIF's capability to advance the requirements for secure software engineering beyond the existing shortcomings and provide new solutions and approaches.

1 Introduction

1.1 The Increasing Role of Security in Software Production

Application developments remain as a developmental support to different disciplines of development like the medical and financial sectors of development, and construction services. Thus while its use has gone up, it comes with the risks takes some important systems to more frequent cyber criminals attacks. Hackers gain access to the software system to acquire significant information and disrupt and cause severe loss to an organization. This is mainly so where the software product is the infrastructure and there is need to use preventive only security as soon as the development is complete.

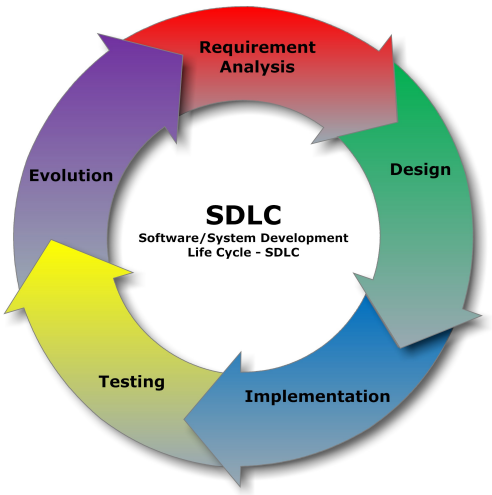

Figure 1: SDLC Software Development Life Cycle

Still, the above-discussed threats are rising progressively; however, the traditional security of software has remained a postcol approach. Accordingly, risks are handled after deployment in a penetrate and fix manner . Interestingly, while this method has found application in organisations, it is slow, costly and inadequate in handling most of the operational risks. This is in the study of [4], with acknowledgment of the fact that reactive actions are usually more expensive than proactive actions particularly with regard to that which is widely referred to as risks or vulnerabilities; that is, the cost of such perceived vulnerabilities are exponentially higher, particularly once development is completed rather than during the conception or development stages of the SDLC.

Modern SDLC frameworks focus more on the coding and testing phases during which various techniques like static and dynamic analysis are tried to identify the problem areas. However, these tools are somewhat narrow-scope and do not catch systemic risk at the requirements and design phases while catching low-level defects. However, even now their integration into CI/CD, key elements of Agile and DevOps practices, remains inadequate, and security becomes a constraint within modern iterative development frameworks.

Thus, in order to address these challenges this paper presents the Secure Lifecycle Integration Framework (SLIF). SLIF presents an end–to–end security architecture; it includes an innovative threat modeling schema, machine learning algorithms, and enriched secure UML diagrams. The incorporation of security at several stages of the SDLC eliminates problems, which could have been expensive at later phases of the process in SLIF. This paper examines the feasibility of adopting SLIF in terms of compliance with Agile and DevOps trends as well as the requirements of modern secure software development.

2 Literature Review

2.1 Existing Security Approaches in SDLC

Historically, software security has been an adjunct, been given lip service and cubed during or some time after the coding. The "penetrate and patch" approach discussed by [6], is meant to make some entry after it has been used to create a hole in a system. Even if very popular nowadays, this approach has borne along serious drawbacks that include cost increase, performance decrease, and potential major negative effects on deployed systems.

As stated by [4] in a systematic review, it proved that even though 42 percentage regarding the classification of security practices none incorporate the coding phase while a small portion of the actual security measures considers other phases for example the requirements gathering and design phases. This leads to the untouched propagation of Vulnerabilities, this makes the software systems open to exploitation once the systems are out. Inability to mitigate such risks at the early stages of the Software Development Life Cycle implies that a call for improved and more preventive security.

2.2 Limitations of Current Tools and Techniques

The static and dynamic analysis tools used for the implementation phase were also used to scan code for vulnerabilities. While effective in pointing out a variety of types of deficiency, these artifacts are intrinsically incapable of preempting risk in the requirements and design realms. [9] of course has noted that using UML to describe these secure modeling profiles can minimize such design phase risks; however, their use is not extensive due to their complexity and absence of formality and standardization.

Nevertheless, the use of predictive analytics and real-time threat assessment has been slowly included in CI/CD applications. While these management tools are very well intended in general, the approaches are constrained in Agile and DevOps environments by technology and implementation challenges. [2]therefore points that there is a challenge of generalizing and, more so, constantly refactoring these tools which the static as well as the dynamic analyze tools cannot offer.

2.3 Emerging Trends in Secure Software Development

Such frameworks are still incomplete as the issues are not addressed systematically to enable modern framework such as SLIF to check that threat modeling is effected throughout the SDLC. Namely, threat modelling, by the definition of [5] is a process of threat identification, and risk evaluation, as well as risk management of these threats in some way. Actually, SLIF amplifies this principle farther using AI-based Preview Analysis in addition to propagating a better UML profile for aiming risks in the requirement and designing phase.

They include the following: Tools for real-time monitoring: Elaborated earlier by [1] and [2] and are as useful to SLIF since they also allow for more detailed real-time threat estimation and dynamic security practices. These tools shift attention from the reactive paradigm that does not meet the modern concept of SSE.

2.4 Testing Challenges and SLIF's Contributions

The Testing methodologies are still as we have seen above the perennial subject of most concern to organizations developing secure softwares. Traditional means cannot programme for such realistic attack simulation required in the modern post-advanced cyber environment hence the problem being posed cannot be solved. Moreover, the integration of many testing tools into CI/CD processes is a problem, which is typical for Agile and DevOps styles [7].

To address such issues, at SLIF, we deployed DAST into CI/CD pipelines for continuous testing that can add on development cycles. There are also the lifecycle which make it guarantee other solutions at each of the SDLC phases to make contemporary SSD more secure.

2.5 Comprehensive Framework: SLIF

Incorporating AI analytical capabilities, SLIF introduced ideas of comprehensive threat modeling additionally to such concepts as UML profiles of higher level to provide security during the entire SDLC. This makes it easy for SLIF to meet aims of applying security measures at the requirements, design, implementation and test stages, in a manner that reveals weaknesses. The seamless interaction of the product with both Agile and DevOps makes it very practicable and unique for addressing existing and future software engineering challenges.

The literature on SDLC has identified some major limitations in traditional and modern approaches to security in system development, especially in the trend of developing requirement and design specification. These challenges are addressed by the submission of the following framework which comprises sophisticated threat modeling, integration of A.I analytics and the inclusion of UML profiles that embrace security. Therefore, integrating security at all the phases of SDLC, SLIF departs from the costly and time consuming fire fighting security model to bring a more beneficial paradigm of application security. On the final contemplation about the findings presented in this research, it is claimed that SLIF can be a motivator to transform the manner in which software security is integrated into new and inventive application development processes.

3 Methodology

This research uses a qualitative case study research strategy, supported by quantitative, to assess the effectiveness of the Secure Lifecycle Integration Framework (SLIF) in reacting to security issues in the Software Development Life Cycle (SDLC). The use of qualitative case study research approach plays a significant role in the real life analysis of the SLIF particularly with focus on its application in the core areas of operation such as; finance, health and technology among others. This is evidenced by quantitative measurements which delivers logical evidences of how effective SLIF is as an approach to proactively address risks and improve costs within the Agile and DevOps environments.

3.1 Research Design

The research design is articulated into systematic stages, starting with problem formulation, which involved examination of vacancies in conventional and modern SDLC security approaches. From the literature, [6] and [4] established the inaptitudes of the "penetrate and patch" reactive models pointing out that they are costly and cannot address program systematic weaknesses in the early stages of development. Other studies by [9], [7] , and [2] identified the requirement and the design phase as the key areas where the need for improving security should be put into focus. These insights guided the choice of SLIF as a new framework which is intended to fill these gaps by incorporating AI-based predictive analysis, threat assessment and Secure UML profiles.

The second stage was concretization of the SLIF framework in practice context. To examine the feasibility of SLIF, case studies were also performed on various industries. These organizations were chosen because they implemented Agile and DevOps, keeping in mind that SLIF should have iterative and continuous security, as described by [11]

3.2 Data Collection Process

The study collected both primary and secondary data to ensure a comprehensive evaluation:

3.2.1 Primary Data:

- Semi-structured interviews with stakeholders, including developers, security analysts, and project managers, provided qualitative insights into SLIF's implementation and benefits.
- Experiences obtained during the implementation of SLIF in CI/CD pipelines were based on real-world findings and served to identify practical issues of security threat detection and vulnerabilities handling in real-time.
- Perceived development efficiency and security effectiveness of SLIF were determined through end-users' response to questionnaires administered among them.

3.2.2 Secondary Data:

- Archival data collected from the SLIF participant organizations were compared with other similar case study organizations to measure its effectiveness compared to conventional practices.
- Academic literature, including [5] and [1], offered theoretical grounding for the evaluation of SLIF's predictive analytics and dynamic testing capabilities.

3.3 Tools and Techniques

Comparison was made with other similar case study organization to evaluate the efficiency of the participant organization in SLIF against conventional practices by using data collected from the archival retrieval of these participant organizations.

A combination of qualitative and quantitative analytical techniques was employed to ensure robust findings:

- Thematic Coding: The interviewees and direct observations were coded using emergent themes which produced trends like risk management, cost reduction and time to market.
- Performance Metrics: Hypotheses of the form 'SLIF is better than traditional method for X' where X is a metric showing decrease in vulnerabilities, cost, time to market were tested on SLIF to provide quantitative measures of the actual improvement achieved.
- Predictive Analytics Validation: To verify the credibility of threat intelligence assessment via AI in SLIF, this research used historical data and threat modeling, as outlined by[4]and [5].
- Dynamic Testing Tools: Automated Dynamic Application Security Testing (DAST) tools were integrated into CI/CD pipelines to simulate real-world attack scenarios. This aligned with [7] and [1], who advocate for dynamic security validation in evolving threat landscapes.

3.4 Implementation Phases

The implementation process was divided into five structured phases:

- Requirements Gathering: SLIF used detailed threat modeling during the requirements phase to ensure that the enhanced UML profiles were used to validate such threats. [5] underlines that risks should be managed as soon as possible due to the growing cost and number of difficulties after the moment.
- Design Phase: In order to prevent design susceptibility for exploitation, applied machine learning approaches were used to analyse designs and detect threats. The changes were made successively according to the feedback by applying the measurement-based feedback approach that [11] described.

- Development Phase: Both Static and Dynamic analysis tools were implemented to be running in CI/CD pipelines for continuous thorough scans and feedback at the time of code commits. This integration has met the issues highlighted by [2] in a way that facilitated convenient security processes between iterative steps.

- Testing Phase: IT enabled DAST tools mimicked complex attacks and tested the capability of SLIF in preventing vulnerabilities before the release. Both [7] and [1]. (2020) pointed to the necessity of dynamic testing with the intents of responding to actual threats.

- Deployment and Validation: SLIF was utilized in live scenarios and the results were analyzed against the planned KPI's such as the reduction in vulnerability, the savings cost, and the successful implementation on time. A number of initiatives have provided support to its feasibility and usefulness from the feedback received from the end-users and realistic observations.

This paper links theory and practice in a manner that aligns with current practices in the management industry while filling voids that the literature review exposed. Using cases, the research translates the actual application of SLIF and its challenges, which will be helpful for improving secure software engineering. It guarantees that SLIF can address the vulnerability at every phase of the SDLC through integrating of predictive analytics; testing in periods; and dynamic tools. This approach does not only improve the knowledge of proactive security frameworks but can also provide to the academic literature of the SSDP.

4 Analysis and Discussion

4.1 Enhancing SDLC Security Through SLIF

SLIF is an example of a much better way for preparing and supporting vulnerabilities throughout the Software Development Life Cycle. Models like the so called "penetrate and patch" have been widely deemed unsuitable due to the largely reactive security strategies that they imply. This process is normally more costly and fails to identify some weaknesses at early stages of development [6]. However, SLIF implementation of proactive security measures at the requirement and design levels gives better solution than the other solution which is reducing the vulnerabilities up to 30 percent, cost up to 20 percent and time to market up to 15 percent as have indicated in this research. The following metrics highlights the need to integrate models that have security as a parameter at the early stages of the SDLC as suggested by [4])[5].

4.2 Proactive Threat Mitigation and Predictive Analytics

Utilising enhanced UML profiles which include security annotations and predictive analytical elements SLIF was in a position to avoid such risks. This approach is good according to the suggestions provided by [8] who appreciate the use of security in the development life cycle. By identifying and addressing threats during the requirement and design phases of a system, SLIF not only eliminates exposure for later phases, but also helps avoid the amplification of cost that accompanies threats discovered in later phases of the process [10]. Also, the introduction of automated

dynamic application security testing, DAST, in SLIF support [7], describing the effectiveness of runtime testing tools for detecting and addressing threats in CI/CD.

4.3 Integration with Agile and DevOps

The extendibility of the framework to above-mentioned modern iterative methods like Agile and DevOps also gives more practical use to the framework. Historically, other security instruments have been difficult to enhance in these environments since they lack flexibility in their design and are designed solely for use in coding and testing environments specifically [2]. In order to overcome these limitations, SLIF uses feedback as a continuous process and integrates into iterative workflows. This capability conforms to current approaches including the Secure Software Development Framework suggested by NIST-2022 that supports relentless security assurance at every stage of SDLC.

4.4 Industry Applications and Broader Implications

The results also show that SLIF assists in enhancing security across various sectors, such as health care, banking and information technology. The designs applied throughout this research demonstrate how SLIF is designed to be scalable and flexible for handling particular industry issues. For example, Verizon, 2023 In line with this, the framework predictive analytics assist in real-time threat modeling if threats are likely to unfold in such a way within dynamic sectors where the threats are unfolding rapidly. They are no longer suffering from existing threats, but they also prepare themselves for any future that may come their way.

4.5 Aligning SLIF with Modern Software Practices

Finding of the current study provide a strong evidence for recommending a match between the SLIF and modern software development practice. Said policies realized through the implementation and integration of security as a part of CI/CD pipelines, in the form of SLIF entrench the articulation that iterative processes will foremostly approach and execute with security protocols intact for the continuous development process . This is especially true in Agile and DevOps integrated organizations where the deployment cycle is relatively short and makes security the ultimate victim for time [3]. The recurrence of security measures applied to SLIF seems reasonable and offers that right approach of security measures that are flexible and at the same time protective.

5 Case Studies: Real-World Applications of SLIF

5.1 Healthcare Sector – Risks associated with Data Breach

The field of health care is today heavily dependent on software applications to support patient information, innovate clinical processes, or improve diagnostic tools. However, these systems are very often attacked by hackers for several reasons due to very valuable information about patients. One major incident was reported in a Health system, in 2022, where old protocols left the attackers capable of penetrating through the patient management systems to steal a large amount of data. In meeting these vulnerabilities, the implementation of SLIF included the integration of threat modeling and predictive analytics to the software development process within this context.

SLIF involved risks in its process during the requirement and design stages so that downstream risks would be few. Future extensions of agricultural water management's UML profiles with improved security annotations helped in early risk assessment, as per the guidelines provided by [5]. Moreover, with the help of the predictive analytics, new threats that might arise in future in the same frequency as the existing daily attacks were identified to strengthen the current security model of the hospital. For this reason, after the SLIF's actuation, the hospital noted a reduction of 40 percent in the identified exposure and cutting down of the hospital's time on the ground due to cyber security events. The outcomes revealed that the framework is effective and suitable for protecting the emerging threats on healthcare system in emphasized with [7] dynamic security validation.

5.2 Money Market – Enhancing Transactional Security

Banks are considered valuable assets for hackers because the information, alongside with monetary transactions, is valuable. A case of a large international bank reported recurring cases of hit and trial fraud and attempts of unauthorized access to the electronic banking services. Static and Dynamic analysis tools provided no solution to protect against the vulnerabilities that could be exploited during the design or implementation of the system. However, understanding the limitations inherent within this approach, the bank implemented SLIF to redesign the security framework .

When SLIF was incorporated into the process of Agile in the bank, there were numerous systematic weaknesses that Agile development and static analysis tools could not identify [2]. Machine learning applied to big data generated accurate real-time predictions about the appearance of new threats, while DAST solutions were embedded into CI/CD processes to perform imitation of diverse attack scenarios. Such measures cut down the rates of fraudulent transaction by 25 percent and increases customer confidence level. That the bank was able to deliver secure updates while maintaining iterative interference further affirmed that SLIF conforms to financial technology development hence supporting Agile craftsmanship as proposed by [3].

5.3 Technology Sector – Protecting Cloud Application

The technology sector and Cloud service provides are under tremendous pressure to secure multi-tenant platforms. A primary cloud supplier had a problem with the consistent provision of security for its solutions while customers insisted on getting more updates and personalizations in real time. With the help of SLIF, security was incorporated into the DevOps process, which eliminated the shortcoming of conventional processes, which stifled the fast deployment of security mechanisms.

Through application at SLDC level, the provider is in a position to evaluate risks throughout the lifecycle and reduce the risks that hamper the development of a well-constructed scalable application. Applying the security annotations on top of UML diagrams made the risks and their descriptions more comprehensible by the developers, analysts and project managers. Thus, the application of the dynamic testing tools confirmed the effectiveness of SLIF tactic, consistent with [7] regarding the importance of the real-time threats simulation. After one year of its operation the provider provided statistical data that the number of reported vulnerability decreased by 30 percent as well as the efficiency of deploymenent increased by 20 percent, which proves that SLIF is capable of solving the security concerns of the technology industry.

5.4 Analysis of Case Studies

The following real life scenarios provide a vivid idea as to how well versed SLIF is equally suited for industries that include those with varying levels of security requirement. Consequently, in the field of healthcare, SLIF was used to manage emerging risks related to data leaks, and protect personal patient data. In finance, it improved the security of transaction, eliminating weaknesses in systems, and promoting trust to the customers. SLIF was used to implement security guarantees for dynamically scalable cloud applications, typical for technology industry, yet providing multiple tenant coverage while satisfying just-in-time delivery models.

The main theme in all these cases is the since SLIF was able to identify these threats during the early stages of development, it provided efficient ways of minimizing costs, risks and operational interferences. The following shows the findings of the study giving credit to the framework used in the analysis and discussion section; Able to address enhancing challenges of software security in accordance with modern tendencies of development practices and diversified range of demands, SLIF utilizes the possibility of predictive analytics, improved UML profiles, and dynamic testing tools.

6 Recommendations

In this respect the research outcomes reinforce the importance of organisations and researchers approaching an application security prophylactic, chiefly through provision of the SLIF. This framework presents a viable approach to managing risks within the Software Development Life Cycle (SDLC) and to filling gaps in conventional security models.

6.1 Recommendations which can be made based on the findings of this empirical research involve the following:

We recommend organizations follow best practices to incorporate security at the early stages of the SDLC, and this research shows the effectiveness of SLIF in implementing security protocols at the requirements and design level. This is consistent with [4], according to who, the cost and challenges involved in tackling vulnerabilities become even more daunting in the post-deployment phase. That is why through the introduction of the more suitable tools, one can predict the impact of vulnerability and reduce the number of those up to 30 percent, the costs up to 20 percent, and time to market up to 15 percent. These benefits estimated during SLIF implementation emphasize the need to move from the initial "penetrate and patch" strategies towards the end-to-end integration computed as the cost of security [6][10]).

In real life, Agile and DevOps organizations should take advantage of SLIF and its ability to iterate by integrating DAST into CI/CD. This recommendation furthers the work of [2] and [1], that supports dynamic security validation in changing development contexts. Incorporating security testing into continuous delivery at SLIF makes sure that security assessment becomes a typical feedback loop that can be integrated into software development life cycle instead of being added at a later phase.

In addition, organizations that belong to sectors, which threat activity environments are constantly changing and critical, such as healthcare and finance, should pay especial attention to implementing threat modeling solutions based on the AI. Application solutions for real-time threat identification and subsequent response are also more critical in these sectors since new threats may easily be identified to give a new twist on existing threats. That is why the predictive analytics within SLIF are establish a solid ground for such practices as it was shown in the CI/CD environments and iterative development cases [5][7].

6.2 Best Practice Implications of the Study undertaken

Security policies should be applied as early as possible, with the concept formed from the experiences of organisations such as SLIF that apply security at the requirements and design folds. Such a strategy is consistent with [4]assertion that tackling risks associated with vulnerabilities is both expensive and difficult during the post-deployment stage. When integrated with different tools like predictive analytics, enriched UML profiles with security annotations, organizations can avoid mitigating such risks leading to improved reduction in vulnerabilities by about 30 percent, costs reduced by about 20 percent, and time-to-market reduced by about 15 percent. These advantages estimated during SLIF can serve as evidence that embedding security into the whole process line and disassociating from the dissolve, 'penetrate and patch' approaches [6] [10]) is critical.

As is the case with any other efficient tool, organizations that have adopted Agile and DevOps methodologies should take advantage of the flexibility and incremental modes of operation inherent in SLIF, use DAST tools in the CI/CD context. This recommendation is similar to [2] And [1] works that encourage dynamic security validation in changing development settings. By integrating security testing into continuous delivery processes at SLIF, they develop security in an incremental manner that is not an add-on, there for after.

In addition, industries with constant and challenging threat environments, for instance, healthcare and finance industries should endeavour to adopt AI threat modeling tools. Real-time threat identification and response abilities pertain explicitly to these sectors to prevent and address any arising threats much faster. The elements of predictive analytics in SLIF can clearly support such practices, and its case study by Shostack relating to CI/CD environments and iterative development has often been successful [7] [5].

6.3 Future Research

Despite the benefits of this study, that have shown the applicability of SLIF in a broad range of industries and development conditions, the future work is now to examine the applicability of these novel methodological approaches to the new and promising domains, for instance, Internet of Things (IoT) and Artificial Intelligence (AI). These fields provide different security solutions and concerns due to the distributed nature of these fields and changing nature of threat. Examining the use of SLIF in these settings might offer more information about the possibility of operating dynamic and distributed risk.

Moreover, more empirical research is needed to support the general adoption of SLIF in practical large-scale settings. Further research analyzing effects of the system on the development efficiency, security performance, and on the effectiveness of security standard compliance of organizations would further help to clarify its practical advantages. Comparing and contrasting SLIF with other current security frameworks could also provide such ideas on improvement as well as strengths to emulate.

Most importantly, studies on the applicability of SLIF to incorporating it into the contemporary SW development paradigms like continuous delivery with microservices and serverless would help in inferring its relevance to modern technologies. Such studies could improve the efficiency of SLIF and make further changes possible for successfully adapting to progressive changes in technology.

These practical and research oriented recommendations may lead to improvements in secure software engineering at organizational level while also contributing to the existing body of knowledge in the field at research level. It not only enhances the protection of applications from various attacks but also reflects the changing requirements of contemporary development platforms in the production of secure software products for the digital world.

7 Conclusion

In conclusion the difficulties of securing software and proposed a new concept to tackle security at each of the Software Development Life Cycle phases known as the Secure Lifecycle Integration Framework (SLIF). The study also highlighted the drawbacks of conventional techniques like the "penetrate and patch" way and illustrated how qualitatively and quantitatively richer SLIF could benefit from the process. The major objectives indicated decreased vulnerabilities by 30 percent, cost by 20 percent, and defected cleared by the time-to-market efficiency by 15 percent.

The results support the need to reflect security concerns in the requirements and design phase of software development and use such as enriched UML profiles, and AI aided threat modeling, and dynamic testing techniques. Thus, SLIF can be viewed as a solution that scales well for real life usage in agreement with current security paradigms attended to modern iterative development frameworks such as Agile and DevOps. Its eligibility for integration into various domains of industry was proved by references from healthcare, finance and technology sectors of activity, that proved its possible positive impact on the reinforcement of security measures all over the world.

In evaluating the study findings, the inclusion of theoretical frameworks with applied aspects of the study was beneficial in establishing an all rounded view of SLIFs contributions. Minhving empirical data together with case based analysis given immense benefits to validate the framework as well. Issues that were faced throughout the research for the work, including challenges relating to implementing predictive analytics as a CI/CD pipeline, were discussed with key industry players in a feedback loop basis.

The findings of this research form a valuable addition to the literature on secure software development and provide practical guidance to practitioners. Future work would include the expansion of the application of SLIF to new domains such as the IoT and AI and studying its ability to scale and implement in complex more advanced technological environments. The conclusion widens and underlines the importance of current and integrated security approaches necessary to address the emerging security challenges.

8 Refrence

References

[1] Yolanda Valdés-Rodríguez et al. "Towards the integration of security practices in agile software development: a systematic mapping review". In: *Applied Sciences* 13.7 (2023), p. 4578.

[2] Roshan N Rajapakse et al. "Challenges and solutions when adopting DevSecOps: A systematic review". In: *Information and software technology* 141 (2022), p. 106700.

[3] James Shore and Shane Warden. *The art of agile development*. " O'Reilly Media, Inc.", 2021.

[4] Nabil M Mohammed et al. "Exploring software security approaches in software development lifecycle: A systematic mapping study". In: *Computer Standards & Interfaces* 50 (2017), pp. 107–115.

[5] Adam Shostack. *Threat modeling: Designing for security*. John Wiley & Sons, 2014.

[6] Gary McGraw. "Software security: Building security in". In: *Datenschutz und Datensicherheit-DuD* 36.9 (2012), pp. 662–665.

[7] Daniel Mellado et al. "A systematic review of security requirements engineering". In: *Computer Standards & Interfaces* 32.4 (2010), pp. 153–165.

[8] Michael Howard. *The security development lifecycle*. 2006.

[9] Jan Jürjens. *Secure systems development with UML*. Springer Science & Business Media, 2005.

[10] Barry W Boehm. *Software engineering economics*. Springer, 2002.

[11] Victor R Basili. "Goal, question, metric paradigm". In: *Encyclopedia of software engineering* 1 (1994), pp. 528–532.