Securing Software Development through People Maturity: A Fuzzy-AHP Decision Making Framework

Hussein A. Al Hashimi , Alaa Omran Almagrabi , Hathal S Alwageed , Ismali M Keshta , Rafiq Ahmad Khan *

Posted Date: 8 July 2024

doi: 10.20944/preprints202407.0669.v1

Keywords: Secure Software Development; Human Success Factors; Human Security Vulnerabilities; Practices, Decision-making framework; Empirical study; Fuzzy-AHP

Article

Securing Software Development through People Maturity: A Fuzzy-AHP Decision Making Framework

Hussein A. Al Hashimi [1], Alaa Omran Almagrabi [2], Hathal S. Alwageed [3], Ismali M. Keshta [4] and Rafiq Ahmad Khan [5,*]

[1] College of Computer and Information Sciences, King Saud University, Saudi Arabia, halhashimi@ksu.edu.sa
[2] Department of Information Systems, Faculty of Computing and Information Technology, King Abdulaziz University, Jeddah, Saudi Arabia, aalmagrabi3@kau.edu.sa
[3] College of Computer and Information Sciences, Jouf University, Sakaka 42421, Saudi Arabia, hswageed@ju.edu.sa
[4] Department of Applied Science, College of Computer Science, Almaarefa University, Riyadh, Saudi Arabia, imohamed@um.edu.sa
[5] Software Engineering Research Group, Department of Computer Science and IT, University of Malakand, Pakistan
* Correspondence: rafiqahmadk@gmail.com

Abstract: In the fast-changing world of software development, the protection of software products has emerged as an imperative requirement. This paper presents a new way to increase the maturity of development teams to reach the highest level of software security. Based on these, the framework uses the Fuzzy Analytic Hierarchy Process (Fuzzy-AHP) to systematically evaluate and enhance the people maturity of software development projects. The essence of the fuzzy logic and AHP technique interaction is to handle the uncertainty and complexity of human factors and team dynamics in the evaluation framework. Using the decision-making model allows the project managers and stakeholders to determine the appropriate areas needed for improvement and develop the right strategies and actions to nurture a secure and mature development culture. The paper identifies 24 human success factors (HSFs) and human security vulnerabilities (HSVs) and 38 practices for addressing these HSFs and HSVs. Furthermore, we discuss the local and global ranks of each HSF and HSV practice and categorize the identified practices into 9 categories to determine the ranks and weight of each category. Based on collected data, FAHP prioritized these practices; the category "C4: Skill Development and Stakeholder Engagement" is ranked highest at rank-1 and possesses the most significant weight of 0.12435. Similarly, the highest global weight is 0.051506, and the global ranked (rank-1) HSF and HSV practice is "P15: Hands-On Practice and Stakeholder Communication". Research evidence and case discussions show how the described framework assists in building secure software development (SSD) practices, which can be considered evidence of the application of team maturity as the means for improving cybersecurity in organizations. Additional research directions include improving the framework, especially using highly developed learning techniques and applying the framework to other forms of development.

Keywords: secure software development; human success factors; human security vulnerabilities; practices; decision-making framework; empirical study; Fuzzy-AHP

1. Introduction

The continuing progress of technology systems has raised the importance of software in various aspects of daily life, including socioeconomic activities. As the reliance on such systems increases, the vulnerability of the software that runs these organizations also increases, making secure software development (SSD) an essential goal for organizations worldwide. These are significant risks associated with security vulnerabilities that are the root causes of financial woes, reputational losses, and unwanted penetration of sensitive data [1]. Therefore, It is evident that software systems' protection is paramount in developing and deploying software systems locally and globally [2].

 doi:10.20944/preprints202407.0669.v1

Recent trends in approaching software security issues involve consideration of the human element in the improved security of the application [3]. Organizational maturity of the development team, best known as people maturity, has been cited as an essential factor that influences the effectiveness of SSD practices. Person maturity includes factors like the behavioural aspect, communication, interaction between the team members, expertise, and experience they possess that either facilitates the security of the software projects or not [4].

However, making people mature and developing such a culture, even in a software development environment, is still tricky [5]. Previous methods of identifying and enhancing team thriving tend to eschew an entirely systematic approach that considers the framework concept when analyzing human factors. Within this context, a research gap is identified that focuses on developing a systematic framework that may be used to systematically assess and facilitate improvement in people's maturity concerning the SSD context. This paper's first and foremost purpose is to bring us to the development of the framework as a guide to help organizations follow a structured approach in assessing the maturity levels of the development team and pinpointing the significant issues. By so doing, it wants to establish a culture of having people continuously learn while adopting changes to ensure that security measures are still effective, given the fast changes and growing technology.

HSFs and HSVs are two sides of the same coin in the context of SSD [6]. People success factors denote characteristics and behaviors that contribute positively to the capabilities of the personnel and subgroups for the development of secure software. These are technical capabilities, communication, problem-solving, security sensitivity, and learning orientation [7]. This paper aims to establish that when these factors are effectively addressed, they play a central role in enhancing secure and dependable software systems.

On the other hand, human security risks arise from the weaknesses and flaws that people and groups may demonstrate, thereby creating threats to software security [6, 8]. Such risks include the user's security knowledge, insufficient training, poor communication, and ignoring security policies [9]. Most insecurity issues likely originate from carelessness, a negligent act, or even deliberate provocation. Hence, it is vital to comprehend and minimize them to strengthen the protective measures of software development projects.

Despite the recognized importance of human factors, there is still a dearth of a framework that systematically examines all of these facets and offers systematic incorporation of these characteristics into the security paradigms of software development [8]. The nature of human interactions and the variability inherent in teamwork complicates defining and identifying all relevant success parameters and potential weaknesses in one complete strategic profile [10].

The human success factors and the security threats have central functions in SSD projects [8]}. The most vital of HSFs are team experience, knowledge, communication, and learning [11]. Expertise is defined as the technical know-how of the personnel in a team assigned to trace and prevent threats to security. This forms part of communication within the team and with stakeholders to ensure the security requirements are understood and implemented [12]. It helps team members to be up-to-date with security trends, thus making it easier to counter them quickly.

On the other hand, security threats result from people’s weaknesses like insufficient training, communication breakdown, and strained cooperation [9]. Lack of training exposes the programmer to a lack of information on new security threats and security measures that can lead to the ease of vulnerabilities in the software being developed [13]. Lack of communication may lead to non-coordination; thus, misinterpretation of the security necessities and inefficient implementation can appear, and inadequate collaboration may cause disintegrated work in securing the software [14]. Also, coding errors and incorrect system configurations contribute to security threats [15]. Mitigating such risks can only be possible by embracing a systems approach in the development of people through acquiring appropriate skills, coordination of efforts in implementing security measures, and effective continuous learning processes to boost the security outcome of software development projects [16].

This paper intends to fill this gap by proposing a comprehensive SSD model that integrates human success factors and security threats. In this way, using methods that consider the complexity

 doi:10.20944/preprints202407.0669.v1

of human factors, such as the fuzzy analytical hierarchy process (Fuzzy-AHP), we will carry out a detailed analysis of the human components of software security to identify ways for improvement. Fuzzy-AHP integrates the multilevel decision-making ability of AHP into the concept of fuzzy, which offers versatility in presenting human factors frequently in vague conditions.

The following sections present precise information on the theoretical background of the proposed framework and explain how it can be used to analyze and improve human factors within the context of software development projects. In this paper, we will demonstrate the implementation and effectiveness of our approach through the literature review supported by empirical data. In addition, we will describe the potential impact of the presented work on the research and practice of software security, and we will stress the importance of equal attention to technical and human factors for attaining holistic software protection.

By doing so, we shall provide pragmatic insights into the call for human factors in SSD and present proposals drawn from the literature for organizations seeking to strengthen their security posture.

The following research questions will help us get closer to our goal:

RQ1: What are the human factors (success factors and security vulnerabilities), as reported in the literature and real-world industry, that influence secure software development projects?

RQ2: Are there any differences between the success factors and security vulnerabilities identified in the literature and real-world industry?

RQ3: What are the best practices, as reported in the literature and real-world industry, that address the human factors (security vulnerabilities) in secure software development projects?

RQ4: How can a decision-making framework be developed to identify the weights of human success factors and security vulnerability practices and evaluate the effectiveness of secure software development projects?

The paper unfolds as follows: Section 2 delves into the motivation and background of the study. A detailed examination of the study's methodology is presented in Section 3. Section 4 conducts an in-depth analysis and evaluation of the acquired results—and section 5 offers securing software development through a people maturity decision-making framework for SSD organizations. Section 6 addresses the study's implications, Section 7 summarizes the study and its limitations, and Section 9 concludes by highlighting avenues for further research.

2. Background and Related Work

This section describes the motivations behind this study, defines software security, and discusses HSFs and HSVs in SSD-related work.

2.1. Motivations Behind the Study

The motivation behind this work is multi-layered, resting on the growing intricacy of software systems and the rising risks that modern enterprises contend with. First, hacking attacks and data leakage are critical issues in software applications and result from poor software development practices [17]. While the traditional security practices and mechanisms are still pertinent, they are insufficient if used individually [18]. It has emerged that project aspects, such as the human facets of development teams and especially their skills, collaborations, and maturity levels, have a significant bearing on the security levels of software systems [19].

Secondly, the current approaches and methods to increase software security mainly use technical disciplines while not considering people's maturity enough [15]. They concluded that there are no well-defined paradigms of human factors' consideration in the decision-making related to SSD [18]. This gap in the current state of knowledge influenced the investigation of a holistic framework to examine and advance the maturity of development teams and, consequently, improve overall security.

Thirdly, the dynamic condition of technology and cyber threats requires decision-making criteria that can be effective in uncertain and changing situations [20]. The Fuzzy Analytical Hierarchy Process (fuzzy-AHP) provides a reliable solution by clearly categorizing AHP and the

 doi:10.20944/preprints202407.0669.v1

elasticity of fuzzy logic [19]. Thus, this hybrid approach enables them to consider other factors of workers' maturity besides the objective since people and technology aspects require such flexibility in their assessment.

Besides, this research appears to be an attempt triggered by the real-world need to encourage organizations to embrace efficient security strategies. In turn, with the assistance of the created framework that enables considering and improving the maturity of development teams systematically, this work will contribute to creating more secure and reliable software systems. The long-term goal is to create a more sustainable approach to software security and encourage the spirit of aspiring to be bigger and better across development teams, decreasing threats to an application's security and improving the software systems' solidness.

In sum, the engineering motivations for undertaking this work are driven by the increasing awareness of the human aspects of software security, the lack of fit of existing paradigms to realize it, and the empirical reality of an organization's struggle to achieve secure software development. Following the proposed Fuzzy-AHP decision-making framework, this research will contribute a valuable framework to improve individuals' maturity and, in turn, increase the security of software development projects.

2.2. Software Security

Software security aims to build programs that can continue to run even when faced with hostile attacks [21]. The best way to reduce software vulnerabilities and problems is to incorporate security and non-functional requirements throughout the Software Development Life Cycle (SDLC) [3]. "Secure software" is an application that follows secure development standards and practices during its design or engineering process [19]. That way, even if the program is attacked, its operations and functions will keep running smoothly. One of the main goals of secure software is to prevent unauthorized individuals from viewing or changing data [22]. Preventing development and maintenance costs requires strict adherence to security standards throughout the design and implementation processes [23].

2.3. People's Maturity towards Secure Software Development

Analyzing the concept of people maturity in the context of secure software development (SSD), it is possible to identify the competency level and the degree of preparedness and activity of individuals and teams regarding security measures within SDLC. Thus, failure to realize high people maturity hinders the development of dependable software systems for responding to emerging threats and risks. Following are the dimensions of people's maturity towards SSD [24-31]:

2.2.1. Dimensions of People Maturity

i. Technical Proficiency and Skill Development: The identification and confirmation that the members of the team have the required competencies is perhaps the most essential element of people's maturity. Enlisting a professional training and education program on the best practices, instruments, and techniques of contemporary securities is also crucial.
ii. Experience: The professional members of the team need to be able to use their expertise and analyze the probable problem related to security and then implement the principles learned most appropriately.
iii. Process Adherence and Standardization: Excellent teams operate with standard methodologies and guidelines that contain security since the beginning of the development stage, for instance, SSDLC models.
iv. Compliance: Thus, following the best practices of the industry and the legally compliant acts guarantees that the security will be the best and most updated.

 doi:10.20944/preprints202407.0669.v1

v. Communication and Collaboration and Interdisciplinary Coordination: Communication is essential because it helps the developers, security specialists, and other related stakeholders to ensure that the security requirements are understood and reflected in the development process.
vi. Feedback Mechanisms: The development of sound feedback mechanisms enables the development team to quickly note or report on security issues as the project progresses.
vii. Proactive Security Culture and Security Awareness: It is compelling to make security a part of a team culture where everyone understands the organization's threats and actively seeks to prevent them.
viii. Responsibility and Accountability: This gives them ownership of security tasks; in the process, individuals feel responsible for the overall security tasks.
ix. Problem-Solving and Analytical Skills: Most well-established teams employ very formal and rational strategies like threat modeling and risk analysis to analyze the weaknesses in security systems.
x. Decision Frameworks: By applying decision-support tools like Fuzzy-AHP, the teams can calculate security measures considering various factors and the fuzziness of the decision-making environment.

2.2.2. Benefits of People Maturity towards Secure Software Development

The following benefits of people's maturity towards SSD are identified through literature [24-31]:

i. Enhanced Security Posture: Originally, higher people's maturity meant a better ability to understand and eliminate security threats, thus improving security maturity.
ii. Reduced Risk: Consequently, the secure maturity level of the formative team means the ability to independently assess the threats and threats handling, which leads to a decrease in the number of and the impacts due to security breaches.
iii. Improved Compliance: Compliance with security standards and meeting regulatory standards increases, meaning the software is legal and complies with industry standards.
iv. Increased Efficiency: That way, the practices are standardized, and communication is efficient in establishing means of providing security without overloading developers for a long time.
v. Continuous Improvement: By practicing open acknowledgement that is not confined to learning through experience but goes further to embrace proactivity, the teams remain informed on the current security trends and hence make constant enhancements to security.

2.2.3. Achieving People Maturity towards Secure Software Development

To achieve people maturity in secure software development, organizations should [24-31]:

i. Invest in Training: Develop and implement continuing formal training sessions and seminars about new means and protection methods.
ii. Implement Standard Processes: First, implement and apply the set of requirements for security across the entire project, not just the separate ones.
iii. Foster a Security Culture: Encourage the organization's personnel, teams, and departments to embrace security matters.

 doi:10.20944/preprints202407.0669.v1

iv. Facilitate Communication: Promoting the free flow of information and patronizing the establishment of rapport between all the participating stakeholders, especially in designing the software.

v. Utilize Decision-Making Tools: Utilize well-formalized decision-making approaches like Fuzzy-AHP in security-related decision-making.

Thus, people maturity is a significant aspect of securing software processes, covering a development team's competencies, practices, information sharing, and decision-making elements. Overall, by methodologically increasing these dimensions, it is possible to create high performance for developing and maintaining secure software systems and, thus, to guard against the constant growth of potential threats in information security.

2.3. Related Work

The concept of people maturity related to secure software development (SSD) has emerged as the focus of attention in recent years. This section presents a literature review and the current frameworks for dealing with human factors in software security. Based on the literature review of previous studies, this section seeks to establish the differences and similarities of the current study regarding their contributions, shortcomings, and the existing research gaps. This research will focus on the following subsections to cover the related work:

2.3.1. Human Factors in Software Security

Human factors, at times, determine the security of software systems. Many papers have stressed the importance of awareness, training, and skills developers must possess to avoid security threats. For instance, a recent study by Acar et al. (2017) assessed the effects of security training on developers' performances in developing secure code, and the authors noted that specific training boosts security solutions. It was also implemented by Xie et al., 2011 the study of the developer's expertise to identify and mitigate security flaws, proving that developed developers can manage security issues.

2.3.2. Secure Software Development Lifecycle

The accommodation of security practices into the Software Development Lifecycle (SDLC) has been popularized for quite some time through several models and frameworks. Another model worth mentioning is the Secure Software Development Lifecycle (SSDLC), which was developed by Microsoft and can be described as the approach that focuses on the idea of security as the integral characteristic of the development process and should be considered and implemented at each stage, starting from the stage of requirements definition and ending with maintenance. This has been found helpful in improving the security of software projects, thus qualifying for improvement. Still, compared with SSDLC, which offers a clear approach, there is not enough emphasis on the human aspect if the security plan is effective.

2.3.3. Maturity Models

Other frameworks and management models, like CMMI and SAMM, are designed to present organizations with checklists to evaluate and advance their processes. These models include aspects that are related to security. Still, many of these models do not inherently address the level of maturity of team members in terms of security. For example, CMMI addresses improving and advancing an organization's capabilities. At the same time, SAMM offers a framework for implementing security, especially in software development, but does not delve into specific skills and team aspects.

2.3.4. Decision-Making Frameworks

AHP and FAHP have been used to solve many decision-making problems in software project management and risk analysis. Fuzzy AHP has mainly been applied to manage the vagueness and subjectivity of human perception. A literature review done by Wang and Elhag (2006) and Lee et al.

 doi:10.20944/preprints202407.0669.v1

(2008) illustrated that Fuzzy AHP is effective in ranking the criteria that cannot be clearly defined with regularity, and the decision-making is made with multitudinous factors under conditions of uncertainty. However, the exact application to measure people's maturity, specifically in secure software development, is still a question mark.

2.3.5. Roles of Human Factors and Security in Integration

Recent studies focus on the interaction of the human element with typical security frameworks. For instance, Assal and Chiasson (2019) developed a framework to examine the effects of various factors attributable to developers in security practices, suggesting that adequate training and awareness should be conducted frequently. Moreover, works such as Green et al. (2016) have also pointed out the need to improve communication and cooperation among the development teams to improve the available security.

2.3.6. Gaps and Opportunities

While the literature on human factors for software development demonstrates an appreciation of people in secure software development, there is a lack of consistent approaches that systematically evaluate and improve people's maturity. Few previous models and frameworks are either technical-oriented or too general and lack a clear structure for measuring the maturity of individuals and teams. Furthermore, there is a lack of research on applying decision-making frameworks such as Fuzzy AHP specifically to health organizations in this context.

2.3.7. Conclusion

Summarizing, it is essential to move a step forward, stressing that even though considerable progress is being made in the acknowledgment of human factors regarding software security, it is crucial to develop a systematic approach to carry out the measurement and improvement of people's maturity within the frame of secure software development. This research will fill this gap using the Fuzzy AHP decision-making approach, which incorporates human factors into the security appraisal. It will offer a single solution for organizations to strengthen security by addressing people's maturity.

3. Research Methodology

To know the impact of identified human success factors (HSFs) and human security vulnerabilities for secure software development (SSD) projects, a three-phase methodology (depicted in Figure 1) was employed. First, a Systematic Literature Review (SLR) identified HSFs and HSVs and recommended practices for SSD projects. Subsequently, an empirical study involving SSD experts was carried out in the second step to assess whether the HSFs, HSVs, and practices identified in the SLR influence the security processes of SSD projects. Finally, the identified HSFs and HSVs practices were ranked using fuzzy-AHP, considering the significance of these practices in the context of the SSD domain.

The outlined phases are further detailed and discussed in the following subsections:

3.1. Phase 1: Systematic Literature Review (SLR)

A systematic literature review (SLR) was conducted initially, and it was used to investigate the realm of HSFs and HSVs and the practices that impact SSD projects. SLRs involve a meticulous and unbiased examination of primary studies, iteratively defining, interpreting, and discussing evidence relevant to research inquiries [32-35]. The guidelines outlined by Kitchenham and Charters [33] were meticulously followed to complete this SLR. SLRs are known to yield more comprehensive and reliable results, as asserted by Kitchenham [36].

SLR's objective is to provide a systematic and comprehensive research study of the literature to evaluate people's maturity contribution to SSD and ensure that potential research gaps are uncovered and addressed effectively while developing the fuzzy-AHP decision-making framework. According to Kitchenham and Charters [33], the following steps are involved in SLR conduction:

 doi:10.20944/preprints202407.0669.v1

3.1.1. Step 1: Research Questions Identification

The first step in designing an SLR involves developing specific and directed questions that the study hopes to answer. In this step, the research questions are clearly stated. It identifies human factors that play crucial roles in the security of software development projects—how people's maturity is measured in the context of SSD.

The following research questions help us get closer to our goals:

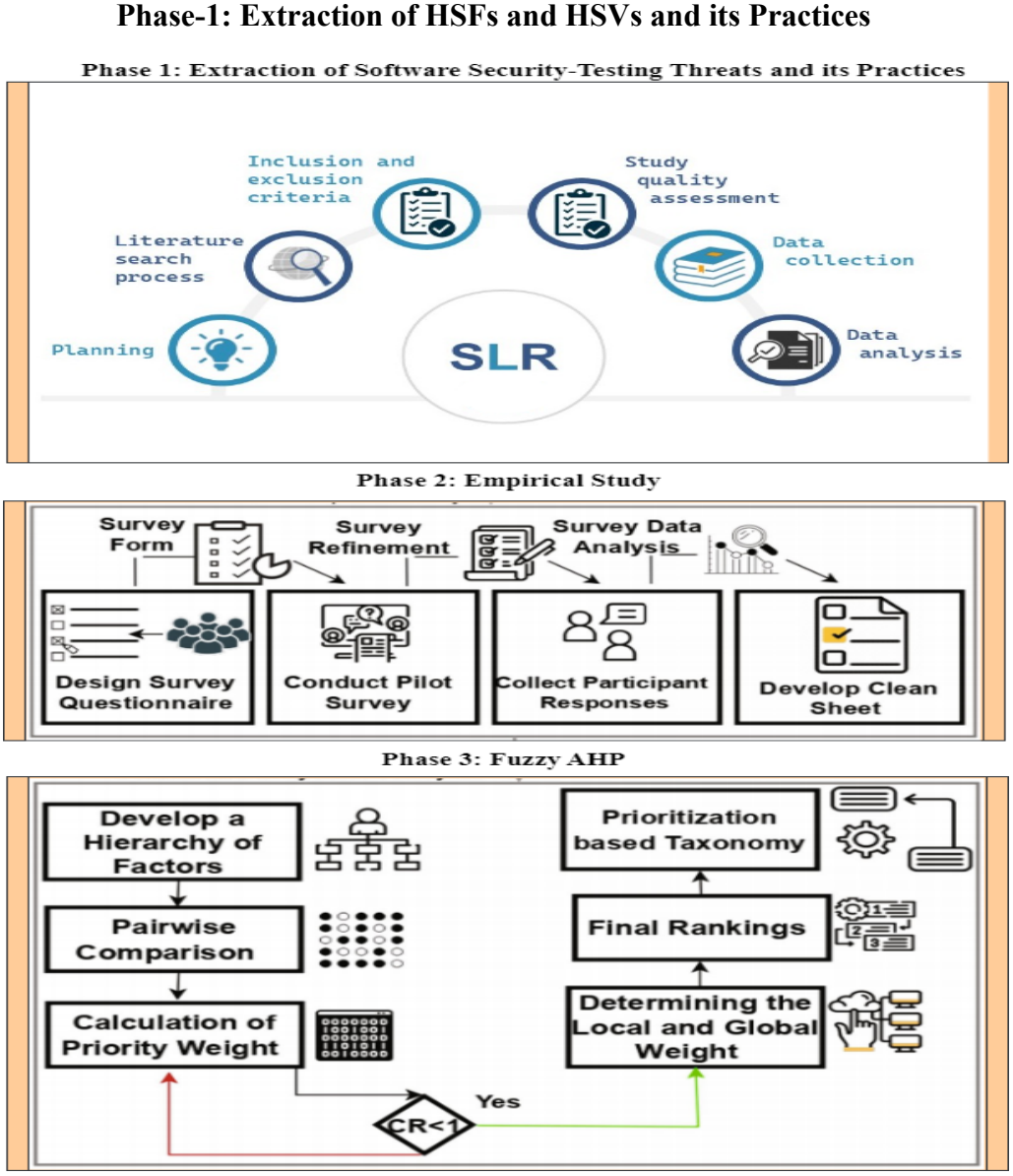

Figure 1. Research Methodology.

RQ1: What are the human factors (success factors and security vulnerabilities), as reported in the literature and real-world industry, that influence secure software development projects?

RQ2: Are there any differences between the success factors and security vulnerabilities identified in the literature and real-world industry?

RQ3: What are the best practices, as reported in the literature and real-world industry, that address the human factors (security vulnerabilities) in secure software development projects?

RQ4: How can a decision-making framework be developed to identify the weights of human success factors and security vulnerability practices and evaluate the effectiveness of secure software development projects?

 doi:10.20944/preprints202407.0669.v1

3.1.2. Step 2: Reviewing or Drafting Review Protocol

Similarly, we create a Review Protocol. A review protocol is, therefore, a plan prepared before carrying out the SLR to determine the most appropriate approach.

- Search Strategy: Specified keywords, databases, and search engines will be used.
- Inclusion and Exclusion Criteria: Selection of literature.
- Data Extraction: The type of data that can be mined from each study.
- Quality Assessment: The quality of the studies included in the review is also considered.

3.1.3. Step 3: Search String Designing and Implementing

The next step we followed was designing and implementing an effective literature search. Here, the following defined search strategy is applied, and a broad search is performed on several databases and search engines. Some possible keywords could be phrases like 'people maturity software security,' 'secure software development,' 'human factor,' 'Fuzzy AHP', and 'decision-making frameworks.'

3.1.4. Step 4: Screening and Selecting Studies

Screening and selecting studies is one of the most critical steps in carrying out a systematic literature review and can take up to two-thirds of the time required to conduct a review. The first step we conducted in the screening process was identifying those studies that might be relevant according to the article's title and abstract. The shortlisted studies are then reviewed in full text to ensure they fit the inclusion/exclusion criteria. Research that examined human aspects that play a role in secure software development. Studies discussing people's maturity from the perspective of software security. Research articles discussing frameworks/ models for evaluating and enhancing people's maturity. Scholarly journals, research conferences, and technical reports which have passed through peer review. This includes all research that did not focus on software development or security. Some of the articles are not indexed for full-text access. Traditional magazines generally do not contain peer-reviewed articles, op-eds,s or editorials. Figure 2 presents the primary and final selection of papers from different digital libraries.

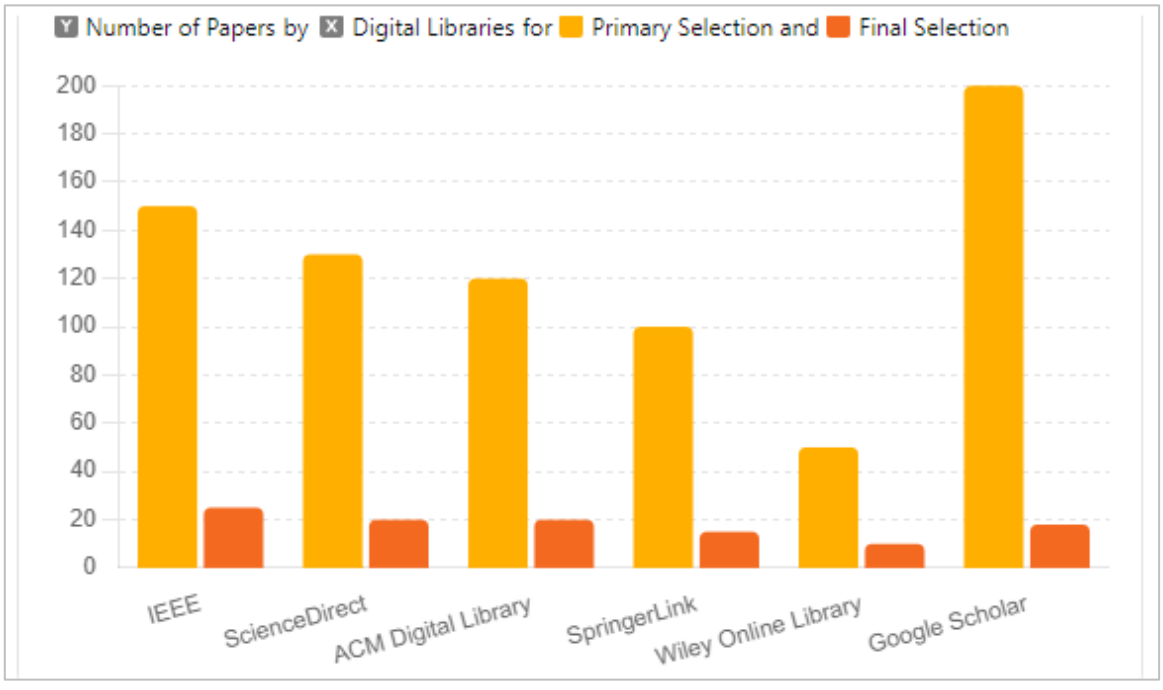

Figure 2. Primary and Final Selection of Digital Libraries.

3.1.5. Step 5: Data Extraction

 doi:10.20944/preprints202407.0669.v1

The information relevant to the review question is collected from the chosen articles using a non-structured questionnaire. The extracted data typically includes:

- Study Title
- Authors
- Publication Year
- Research Objectives
- Methodology
- Key Findings
- Limitations
- Recommendations for Future Research

3.1.6. Step 6: Quality Assessment

Quality appraisal of each selected study is done using specific quality criteria. Common quality assessment criteria include:

- Clarity of research objectives
- Rigor of the methodology
- Internal and external validity of the research
- Contribution to the field
- Honesty regarding certain constraints and perspectives

3.1.7. Step 7: Data Synthesis

In addition, patterns, themes, and relations between the studies are established from the extracted data. This refers to the number of themes identified and the style of making the synthesis where one provides narrative synthesis. Concerning the research questions, the findings are posed systematically.

3.1.8. Step 8: The Findings

The results of the SLR are compiled into a detailed report structured in Section 4 of this study.

Thus, by following these simple and structured steps, the literature review will help gain a thorough appreciation of people's maturity and significance in secure software development, help identify the missing links, and help develop a perfect and comprehensive Fuzzy-AHP decision-making system.

3.2. Phase 2: Empirical Study

We performed an empirical study using an online survey to gather insights from SSD experts and practitioners. The focus was on their experiences handling HSFs and HSVs and their practice's impact on SSD projects.

Employing an online survey in this study offered several advantages [37]:

- This method eliminated the necessity for scheduling meetings or global travel, allowing for efficient data collection from experts across continents.
- Utilizing online resources that are both cost-effective and accessible, such as Google Online Form, made the implementation process more practical.

 doi:10.20944/preprints202407.0669.v1

The survey had design and sampling phases. During the design phase, several systematic and unsystematic sampling methods were considered when formulating sample questions [38]. We chose an online survey, a non-scientific data collection method often used for information gathering by other researchers [39-41] because it was more practical than directly collecting data from experts in different countries.

3.2.1. Development of Questionnaire Survey

HSFs and HSVs were evaluated in the closed-ended section using a five-point Likert scale, ranging from 'strongly agree' to 'strongly disagree.

3.2.2. The Pilot of Questionnaire Survey

To evaluate the questionnaire survey, a pilot study involved experts from software development organizations, namely, the "Cyber-Physical Systems Research Group, University of Southampton, UK", "College of Computer and Information Sciences, King Saud University, Saudi Arabia", "Software Engineering Research Group (SERG_UOM) Pakistan". Subsequently, expert feedback was incorporated, leading to revisions in the questionnaire survey.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 8 July 2024 doi:10.20944/preprints202407.0669.v1

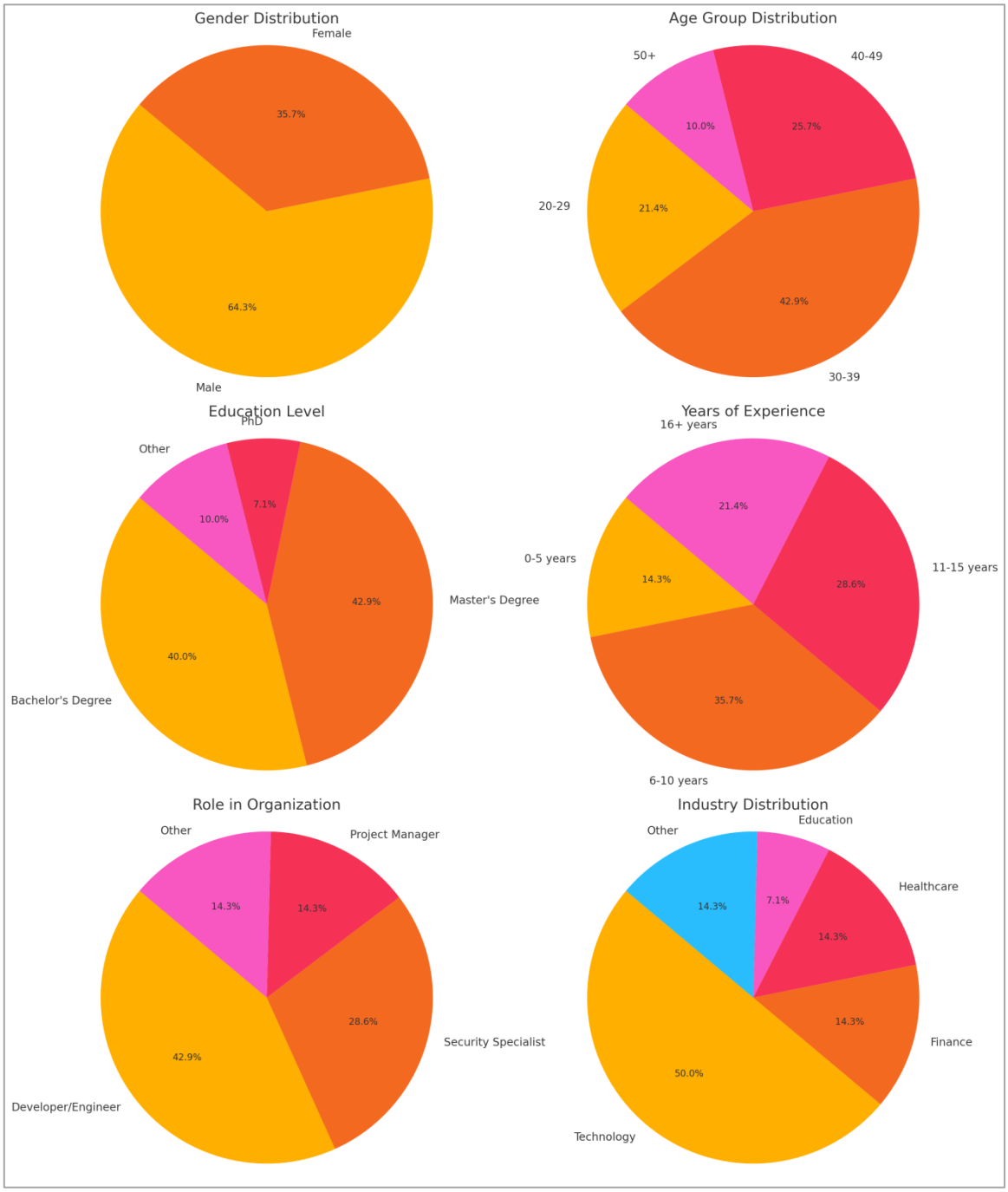

Figure 3. Demographic details of survey participants.

3.1.3. Data Collection Sources

Utilizing snowball sampling, expert data was gathered [42]. Email and numerous social media sites, such as ResearchGate, Facebook, LinkedIn, and Gmail, were used to make contact. Out of 90 responses received from SSD survey participants, 20 were excluded as their expertise didn't align with secure software development. Subsequently, the final dataset of 70 survey responses underwent manual evaluation and was used for subsequent statistical analysis. Respondents predominantly held Master's degrees, while those with PhDs were less represented. Most participants hailed from prominent SSD enterprises, with many holding decision-making roles.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 8 July 2024 doi:10.20944/preprints202407.0669.v1

Figure 3 presents the demographic details of the 70 survey participants for identifying human success factors (HSFs) and human security vulnerabilities (HSVs) that impact secure software development (SSD) projects.

3.3. Phase-3: Fuzzy Analytical Hierarchy Process (FAHP)

The list of HSF and HSV practices was ranked based on the findings of the fuzzy analytical hierarchy process (Fuzzy-AHP).

3.3.1. Fuzzy Set

This section provides an in-depth discussion of fuzzy set theory and fundamental AHP concepts. Initially introduced by Zadeh [43], fuzzy set theory was devised to address uncertainties and vagueness inherent in real-world problems. Its main benefit is representing imprecise data [44]. Within a fuzzy set, membership functions map objects onto a scale between '0' and '1'.

Definition: In Figure 4, a triangular fuzzy number (TFN) denoted as F comprises a set (fl, fm, fu), and its membership function μF (x) is defined by equation (1).

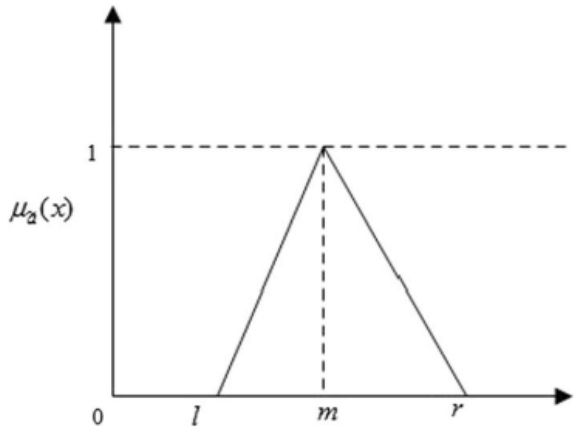

Figure 4. Triangular Fuzzy Number.

$$\mu_F(x) = \begin{cases} \frac{x-f^l}{f^m-f^l}, & f^l \le x \le f^m \\ \frac{f^u-x}{f^u-f^m}, & f^m \le x \le f^u \\ 0, & otherwise \end{cases} \qquad (1)$$

when the exact values for the lowest, most significant, and highest ranges are denoted by (fl, fm, fu). Triangular fuzzy numbers (TFNs) for T1 and T2 are mathematically represented in Table 1.

Table 1. Triangular Fuzzy Numbers.

Operation Laws	Expression
Addition ($F_1 \otimes F_2$)	$(f^l_1, f^m_1, f^u_1) \otimes (f^l_2, f^m_2, f^u_2) = (f^l_1 + f^l_2, f^m_1 + f^m_2, f^u_1 + f^u_2)$
Subtraction ($F_1 \otimes F_2$)	$(f^l_1, f^m_1, f^u_1) \otimes (f^l_2, f^m_2, f^u_2) = (f^l_1 - f^l_2, f^m_1 - f^m_2, f^u_1 - f^u_2)$
Multiplication ($F_1 \otimes F_2$)	$(f^l_1, f^m_1, f^u_1) \otimes (f^l_2, f^m_2, f^u_2) = (f^l_1 * f^l_2, f^m_1 * f^m_2, f^u_1 * f^u_2)$
Division ($F_1 \otimes F_2$)	$(f^l_1, f^m_1, f^u_1) \otimes (f^l_2, f^m_2, f^u_2) = (f^l_1 / f^l_2, f^m_1 / f^m_2, f^u_1 / f^u_2)$
Inverse ($F_1 \otimes F_2$)	$(f^l_1, f^m_1, f^u_1)^{-1} = (1/f^l_1, 1/f^m_1, 1/f^u_1)$
For any real number k (Kf_1)	$k\,(f^l_1, f^m_1, f^u_1) = kf^l_1, kf^m_1, kf^u_1$

3.3.2. FAHP

 doi:10.20944/preprints202407.0669.v1

The Analytical Hierarchy Process (AHP) is used to solve "multi-criteria decision-making" (MCDM) problems. The limitations of conventional AHP methods, such as a "crisp environment," an absence of uncertainty consideration, unbalanced judgmental scales, and subjective judgment selection, have been greatly helped by AHP and fuzzy set theory. Thus, in MCDM scenarios involving uncertainties and fuzziness, the Fuzzy Analytical Hierarchy Process (FAHP) has become increasingly popular [45]. Using the Triangular Fuzzy Numbers (TFNs) scale, FAHP allows linguistic terms to be incorporated to collect input from several decision-makers. This provides the opportunity to quantify the linguistic variables. Similar techniques have been utilized to gauge vagueness in fuzzy environments across various engineering disciplines [46]. This study adopts Chang's FAHP [47], recognized for its consistency and appropriateness in such analyses.

Imagine a situation where you must prioritize HSF and HSV practices toward SSD projects. There are two sets, X = {x1', x2', … xn} and U = {u1', u2', … un}, where X is the set of objects and U is the set of goals. Chang [47] asserts that each object undergoes measurement while every goal (gi) is pursued. By employing equations (11) and (12), the extent of analysis values (m) for each object can be determined.

$$\widetilde{F^1}_{gi} + \widetilde{F^2}_{gi} , \dots \widetilde{F^m}_{gi} \qquad (2)$$

$$i = 1, 2, \dots, n \qquad (3)$$

where the TRNs indicate $\widetilde{F^j}_{gi}$ (where j=1, 2, …, m). The following procedures are carried out to apply Chang's extent analysis methodology:

Step 1: The first step is to create a comparison matrix that considers fuzzy values.

Step 2: Figure out the fuzzy synthetic extent as it relates to the options

$$Si = \sum_{j=1}^{m} \widetilde{F^j}_{gi} \otimes \left[\sum_{i=1}^{n}\sum_{j=1}^{m} \widetilde{F^j}_{gi}\right]^{-1} \qquad (4)$$

To achieve the expression $\sum_{j=1}^{m} \widetilde{F^j}_{gi}$, "execute the fuzzy addition operation of m extent analysis such as:"

$$\sum_{j=1}^{m} \widetilde{F^j}_{gi} = \left(\sum_{j=1}^{m} \widetilde{f^l}_{gi}, \sum_{j=1}^{m} \widetilde{f^m}_{gi}, \sum_{j=1}^{m} \widetilde{f^u}_{gi}\right) \qquad (5)$$

and to achieve the expression $\left[\sum_{i=1}^{n}\sum_{j=1}^{m} \widetilde{F^j}_{gi}\right]^{-1}$, "the fuzzy addition operation is executed on" $\widetilde{F^j}_{gi}$(j = 1, 2, ...m) value, as follow:

$$\sum_{i=1}^{n}\sum_{j=1}^{m} \widetilde{F^j}_{gi} = \left(\sum_{i=1}^{n} \widetilde{f^l}_{i}, \sum_{i=1}^{n} \widetilde{f^m}_{i}, \sum_{i=1}^{n} \widetilde{f^u}_{i}\right) \qquad (6)$$

Finally, calculate the vector's inverse using Eq. (7):

$$\left[\sum_{i=1}^{n}\sum_{j=1}^{m} \widetilde{F^j}_{gi}\right]^{-1} = \left(\frac{1}{\sum_{i=1}^{n} \widetilde{f^u}_{i}}, \frac{1}{\sum_{i=1}^{n} \widetilde{f^m}_{i}}, \frac{1}{\sum_{i=1}^{n} \widetilde{f^l}_{i}}\right) \qquad (7)$$

Step 2: "As F_a and F_b are two triangular fuzzy numbers, then the degree of possibility of" F_a= (f_a^l, f_a^m, f_a^u) ≥ F_b= (f_b^l, f_b^m, f_b^u) is defined as follows. The Equation (8) is also given below:

$$V(F_a \geq F_b) = \sup\ [\min(\mu_{Fa}(x), \mu_{Fb}(x))] \qquad (8)$$

$$V(F_a \geq F_b) = \text{hgt}\ (F_a \cap F_b) = \mu_{Fa}(d) = \begin{cases} 1, & if\ f_a^m \geq f_b^m \\ \frac{f_a^u - f_b^l}{(f_a^u - f_b^m) + (f_b^m - f_b^l)}, & f_b^l \leq f_a^u \\ 0, & otherwise \end{cases} \qquad (9)$$

Here, the ordinate of the highest point where d, μ_Fa, and μ_Fb intersect is represented by the value of d (Figure 5).To find out what P1 and P2 are worth, you need to know the values of V1(F_a≥ F_b) and V2(F_a≥ F_b).

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 8 July 2024 doi:10.20944/preprints202407.0669.v1

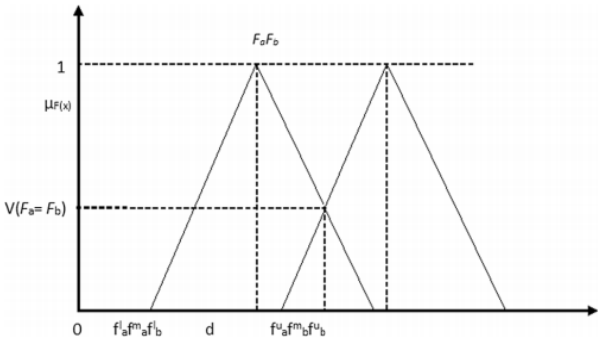

Figure 5. Intersection between Triangular Fuzzy Numbers.

Step 3: The following is one way to express the process of finding the full "degree of possibility of a convex fuzzy number and other convex fuzzy numbers Fi (i=1, 2,…, k)":

$$V(F \geq F1, F2, F3, \ldots Fk) = \min V(F \geq Fi) \quad (10)$$

Assuming:

$$d/Fi = \min V(Fi \geq Fk) \quad (11)$$

for k=1,2,…,n; k≠i.

It is through the utilization of Equation 21 that the weight vector is established.

$$W' = (d'(F_1), d'(F_2), d'(F_3) \ldots, d'(F_n)) \quad (12)$$

in which Fi are definite variables where i=1,2,...,n.

Step 4: The weight vector obtained from Equation (14) and Equation (13) standardizes it. This normalized non-fuzzy value indicates priority weight for HSF and HSV practices.

$$W = (d(F_1), d(F_2), d(F_3) \ldots, d(F_n)) \quad (13)$$

with W standing for the weight of a security risk's priority.

Step 5: Fifthly, ensure that the matrices used to compare options in fuzzy AHP are consistent [48]. The Consistency Ratio (CR) calculation for each matrix is an essential component. Fuzzy matrices can be created using the graded-mean method. The triangular fuzzy matrix P = (l, m, u) is transformed using Equation 14:

$$P_{crisp} = \frac{(4m+l+u)}{6} \quad (14)$$

The final consistency ratio is calculated using equations (15) and (16):

$$CI = \frac{(\lambda_{max} - n)}{n-1} \quad (15)$$

$$CR = \frac{CI}{RI} \quad (16)$$

Where λ_{max} represents "the largest eigenvalue of the comparison matrix", n is "the number of items being compared in the matrix," and RI is "the random index. Its value can be referenced from Table 2." CI is "the consistency index" computed using equation (16). The matrix maintains consistency if CR is below 0.1; otherwise, decision-makers should resort to pair-wise judgments.

Table 2. Random Consistency Index (RI) concerning matrix size.

Matrix size	1	2	3	4	5	6	7	8	9	10
RI	0	0	0.58	0.9	1.12	1.24	1.32	1.41	1.45	1.49

4. Results and Data Analysis

In this section, we describe the detailed analysis of the data acquired through the empirical survey, and FAHP used to answer the research questions (RQs) posed in Section 1:

 doi:10.20944/preprints202407.0669.v1

4.1. RQ1: What are the Human Factors (Success Factors and Security Vulnerabilities), as Reported in the Literature and Real-World Industry, that Influence Secure Software Development Projects?

Several human factors significantly influence success in secure software development (SSD) projects. These factors encompass various aspects of team dynamics, knowledge, communication, and organizational culture. Similarly, in SSD projects, various human factors can introduce security vulnerabilities that compromise the software's integrity, confidentiality, and availability. Understanding these factors is crucial to mitigating risks and ensures the SSD process.

Table 3 presents the key human factors (success factors and security vulnerabilities) that impact SSD projects:

Table 3. Human Factors (Success Factors and Security Vulnerabilities) as identified in the literature and real-world industry that influence SSD Projects.

Human Success Factors (HSFs)	Description		Human Security Vulnerabilities (HSVs)	Description	
HSF1: Skill and Expertise	**Technical Proficiency:** The team must possess strong technical skills in software development, cybersecurity principles, and secure coding practices. Expertise in using security tools and technologies is also crucial.	**Continuous Learning:** Encourage constant learning and professional development to keep team members updated with the latest threats, vulnerabilities, and mitigation techniques.	**HSV1: Lack of Security Awareness and Training**	**Insufficient Knowledge:** Developers and other team members who lack proper training in secure coding practice and Cybersecurity principles are more likely to introduce vulnerabilities into the software.	**Outdated Knowledge:** Not staying updated with the latest security threats, trends, and best practices can result in obsolete and insecure methods.
HSF2: Awareness and Mindset	**Security Awareness:** Every team member, from developers to project managers, should know the importance of security in the SDLC. Regular training and awareness programs can help reinforce this mindset.	**Security-First Mindset:** Cultivating a security-first mindset means prioritizing security considerations in every phase of the SDLC. This involves integrating security practices into the development process rather than the team treating them as an afterthought.	**HSV2: Poor Communication and Collaboration**	**Miscommunication:** Inadequate communication between team members, especially between developers and security experts, can lead to misunderstandings and overlooked security requirements.	**Siloed Teams:** Lack of collaboration between teams (development security, QA, operations) can result in incomplete security reviews and unaddressed vulnerabilities.
HSF3: Communication and Collaboration	**Effective Communication:** Clear and open communication channels within the team and with stakeholders are essential. This	**Collaboration:** Cross-functional collaboration between developers, security experts, quality assurance teams, and operations staff is	**HSV3: Inadequate Testing and Review**	**Insufficient Security Testing:** Failing to perform comprehensive security testing, such as static code analysis, dynamic testing, and	**Lack of Peer Reviews:** Skipping code reviews or conducting superficial reviews can allow security

 doi:10.20944/preprints202407.0669.v1

	ensures that security requirements, risks, and mitigation strategies are well understood and properly implemented.	crucial. Collaborative efforts help identify potential security issues early and address them efficiently.		penetration testing, can leave vulnerabilities undetected.	flaws to persist in the codebase.
HSF4: Leadership and Management	**Supportive Leadership:** Leaders and managers should support and advocate for secure development practices. This includes allocating necessary resources, providing training opportunities, and emphasizing the importance of security in project goals.	**Risk Management:** Effective leadership involves identifying, assessing, and managing security risks throughout the project. Proactive risk management helps mitigate potential threats before they become critical issues.	**HSV4: Pressure and Workload**	**Time Constraints:** Tight deadlines and high workload pressures can lead to cutting corners, skipping security checks, and hastily implementing code that may contain vulnerabilities.	**Burnout:** Overworked and fatigued developers are more prone to making mistakes and overlooking critical security.
HSF5: Process and Methodology	**Defined Processes:** Establishing well-defined processes and methodologies for SSD, such as Secure Development Lifecycle (SDL) or DevSecOps, helps ensure that security practices are systematically followed.	**Compliance and Standards:** Adhering to industry standards and regulatory requirements, such as ISO/IEC 27001, NIST, or GDPR, ensures that security practices are aligned with best practices and legal obligations.	**HSV5: Unclear Roles and Responsibilities**	**Ambiguous Accountability:** When security responsibilities are unclear, important security tasks may be neglected or assumed to be someone else's responsibility.	**Lack of Ownership:** Without clear ownership of security tasks, there may be a lack of accountability and commitment to secure coding practices.
HSF6: Culture and Environment	**Security Culture:** Building a security-centric culture within the organization promotes vigilance and accountability. A strong security culture encourages team members to identify and address security concerns proactively.	**Psychological Safety:** Creating an environment where team members feel safe to report security issues without fear of blame or retribution fosters openness and prompt resolution of potential vulnerabilities.	**HSV6: Resistance to Change**	**Inertia:** Resistance to adopting new security tools, practices, or frameworks due to comfort with existing methods can hinder the implementation of more secure practices.	**Fear of Disruption:** Concerns about disrupting existing workflows or delaying projects can lead to resistance against integrating security measures into the development process.

 doi:10.20944/preprints202407.0669.v1

HSF7: Accountability and Responsibility	**Clear Roles and Responsibilities:** Clearly define responsibilities related to security within the team to ensure that everyone knows their part in maintaining and enhancing software security.	**Ownership:** Encouraging team members to take ownership of their work and its security implications leads to higher quality and more SSD outcomes.	**HSV7: Insufficient Resources**	**Limited Budget:** Inadequate funding for security training, tools, and resources can prevent the adoption necessary security measures.	**Lack of Access to Expertise:** Not having access to experienced security professionals for guidance and support can leave security gaps unaddressed.
HSF8: Resource Allocation	**Adequate Resources:** Providing sufficient resources, including time, budget, and tools, is essential for implementing and maintaining SSD practices.	**Access to Expertise:** Ensuring access to security experts, whether in-house or through external consultants, helps address complex security challenges effectively.	**HSV8: Human Error**	**Coding Mistakes:** Simple coding errors, such as improper input validation, hardcoded credentials, or incorrect configurations, can introduce significant vulnerabilities.	**Configuration Errors:** Misconfiguratio ns in development environments, servers, and applications can create exploitable security gaps.
HSF9: Testing and Validation	**Comprehensive Testing:** Regular and thorough security testing, including code reviews, penetrating testing, and vulnerability assessments, helps identify and rectify security weaknesses.	**Feedback Loops:** Establishing feedback loops for continuous improvement ensures that lessons learned from past projects are applied to future endeavors, enhancing overall security practices.	**HSV9: Neglecting Secure Development Lifecycle**	**Inconsistent Processes:** Failing to adhere to a standardized, secure development lifecycle (SDL) can result in inconsistent application of security practices and unaddressed vulnerabilities.	**Ignoring Best Practices:** Not following established best practices for secure development can introduce common vulnerabilities.
HSF10: Incident Response and Recovery	**Preparedness:** A well-defined incident response plan ensures the team is prepared to handle security breaches effectively. This includes identifying, responding to, and recovering from security incidents.	**Learning from Incidents:** Analyzing and learning from security incidents helps improve security measures and prevent future occurrences.	**HSV10: Cultural Issues**	**Lack of Security Culture:** An organizational culture that does not prioritize security or view it as everyone's responsibility can lead to neglect of security practices.	**Blame Culture:** A culture that penalizes individuals for reporting security issues can discourage team members from identifying and addressing vulnerabilities.
HSF11: Scalability and Flexibility	**Adaptable Integration:** Ensuring that	**Scalable Solution:** Develop solutions that can scale up to	**HSV11: Insufficient**	**Lack of Preparedness:** Inadequate	**Failure to learn from Incident:** Not analyzing

 doi:10.20944/preprints202407.0669.v1

	secure development practices can be integrated into various healthcare settings, from small clinics to large hospitals, and tailored to meet specific needs.	handle increased volumes of interactions and data while maintaining security standards.	**Incident Response Preparedness**	incident response planning and training can result in poor handling of security incidents, leading to prolonged exposure to vulnerabilities.	and learning from past security incidents can result in recurring vulnerabilities and security breaches.
HSF12: Ethical and Cultural Sensitivity	**Ethical Considerations:** Addressing ethical considerations comprehensively ensures patient confidentiality, informed consent, and equitable access to new interventions.	**Cultural Sensitivity:** Recognizing and respecting cultural differences in healthcare practices and patient expectations enhances acceptance and effectiveness of secure software.	**HSV12: Over-Reliance and Tools**	**Tool Dependency:** Relying too heavily on automated security tools without human oversight can lead to missed vulnerabilities that require contextual understanding and manual intervention.	**False Sense of Security:** Assuming that using security tools alone is sufficient can lead to complacency and underestimation of the need for thorough security practices.

By focusing on these human success factors (HSFs) and human security vulnerabilities (HSVs), organizations can significantly enhance the success of their SSD projects, resulting in more resilient and trustworthy software solutions.

4.2. RQ2: What are the Similarities and Differences between HSFs and HSVFs, as Identified through Literature and Real-World Industries, that Influence SSD Projects?

To analyze the similarities and differences between human success factors (HSFs) and human security vulnerabilities factors (HSVFs) as identified through literature and real-world industries that influence secure software development (SSD), we considered how these factors and vulnerabilities are perceived and addressed in both contexts (as presented in Table 4 and 5).

Table 4. Similarities and Differences of HSFs identified through Literature and Real-World Industries.

HSFs	Literature	Real-World Industries	Similarities	Differences
HSF1: Skill and Expertise	Emphasizes the need for technical proficiency and continuous learning	Highlights the importance of training and identifying skill gaps in practical scenarios	Both stress the importance of technical skills and ongoing education	The literature review focuses more on the ideal skill sets and continuous learning, while surveys highlight real-world skill deficiencies
HSF2: Awareness and Mindset	Stress the importance of security awareness and a security-first mindset throughout the SDLC.	Indicates a lack of security awareness as a common issue among practitioners	Both emphasize the critical role of awareness and mindset in ensuring security.	Literature review emphasizes theoretical importance, while surveys highlight practical shortcomings in awareness and mindset.
HSF3: Communication and Collaboration	Highlights the importance of clear communication and	Identifies poor communication and collaboration as	Both recognize that effective communication and	The literature review discusses ideal communication strategies, while surveys focus on

 doi:10.20944/preprints202407.0669.v1

	cross-functional collaboration	significant issues impacting security	collaboration are essential for security.	practical communication breakdowns.
HSF4: Leadership and Management	Discusses the role of supportive leadership and proactive risk management	Often highlights a lack of clear leadership and accountability in practice	Both agree on the importance of solid leadership and management in promoting security	The literature review provides theoretical frameworks for leadership, while surveys offer specific examples of leadership deficiencies
HSF5: Process and Methodology	Advocates for defined processes and adherence to secure development lifecycles	Points out the neglect of secure development lifecycles and methodologies in practice	Both stress the necessity of structured processes and methodologies for security.	The literature review focuses on ideal processes, while the survey highlights the practical neglect of these processes.
HSF6: Culture and Environment	Emphasizes the need for a security-centric culture and supportive environment	Identifies cultural issues and resistance to change as significant barriers to security	Both recognize the influence of organizational culture on security practices.	The literature review discusses theoretical and cultural frameworks, while surveys address specific organizational cultural challenges.
HSF7: Accountability and Responsibility	Stresses the importance of clear roles and responsibilities in maintaining security	Highlights the impact of unclear roles and responsibilities on security practices in real-world scenarios	Both emphasize the need for clear accountability to ensure security	A literature review may discuss theoretical role definition, while surveys highlight real-world ambiguities and their impact on security
HSF8: Resource Allocation	Discusses the need for adequate resources, including time, budget, and tools, for secure development	Identifies insufficient resources as a common issue affecting security efforts	Both stress the importance of resource allocation for maintaining security	The literature review focuses on ideal resource allocation, while surveys highlight practical
HSF9: Testing and Validation	Advocates for thorough security testing and validation processes throughout the development lifecycle	Points out inadequate testing and review as significant vulnerabilities in practice	Both agree on the importance of comprehensive testing and validation	Literature review details ideal testing methodologies, while surveys reveal practical deficiencies in testing and validation efforts
HSF10: Incident Response and Recovery	Emphasizes the need for preparedness and learning from security incidents to improve future responses	Highlights insufficient incident response preparedness and real-world challenges in handling incidents	Both stress the importance of having robust incident response and recovery plans.	The literature review discusses theoretical incident response plans, while surveys highlight practical gaps and challenges in preparedness.
HSF11: Scalability and Flexibility	Discuss the need for adaptable and scalable security solutions that can grow with the organization.	Often implied in resource allocation and process flexibility but not always explicitly addressed.	Both recognize the need for scalability and flexibility in security practices.	The literature review explicitly addresses scalability and flexibility, while the survey focuses more on immediate practical concerns.
HSF12: Ethical and Cultural Sensitivity	Highlights the importance of ethical considerations and cultural sensitivity in SSD	Often implied in broader cultural issues and security awareness but not always explicitly addressed.	Both touch upon the importance of considering ethical and cultural factors in security practices	Literature review explicitly addresses ethical and cultural frameworks, while surveys focus on immediate practical concerns.

 doi:10.20944/preprints202407.0669.v1

Table 5 summarizes the similarities and differences between human security vulnerabilities (HSVs) identified through literature and those observed in real-world industries, clearly comparing common themes and unique challenges faced in both contexts.

Table 5. Similarities and Differences of HSVs Identified through Literature and Real-World Industries.

HSVs	Literature	Real-World Industries	Similarities	Differences
HSV1: Lack of Security Awareness and Training	Emphasizes the importance of continuous education and awareness programs	Highlights frequent security breaches due to sufficient knowledge	Both stress the need for awareness and training in security practices	Literature focuses on theoretical frameworks, while the real world highlights practical training gaps
HSV2: Poor Communication and Collaboration	Academic studies underline the importance of clear communication and collaboration.	Industries experience issues where a lack of collaboration leads to security gaps	Both recognize the importance of effective communication and collaboration	Literature provides ideal communication strategies, while the real world highlights practical communication issues
HSV3: Inadequate Testing and Review	Security vulnerabilities often arise from inadequate testing and review processes.	Time constraints and resource limitations result in rushed or incomplete testing.	Both stress the importance of comprehensive testing and validation	Literature details ideal testing methodologies, while the real world highlights practical deficiencies of testing efforts
HSV4: Unclear Roles and Responsibilities	Research identifies the need for clearly defined roles and responsibilities to avoid security issues.	Unclear responsibilities can lead to essential security tasks being overlooked or improperly executed.	Both agree on the importance of clearly defined roles and responsibilities in promoting security.	Literature provides theoretical frameworks for roles, while the real world highlights practical issues related to unclear roles.
HSV5: Insufficient Resources	Emphasizes the need for adequate resources, including budget, tools, and personnel	Budget constraints often lead to prioritizing other operational needs over security investments.	Both stress the importance of resource allocation for security	Literature focuses on ideal resource allocation, while the real world highlights practical resource constraints
HSV6: Human Error	Academic work acknowledges human error as a critical factor in security breaches.	Human error is a common cause of security incidents in industry reports	Both agree on the importance of addressing human error in security practices	Literature provides theoretical approaches, while the real world highlights practical human error issues
HSV7: Pressure and Workload	Less commonly highlighted compared to other vulnerabilities	High pressure and workload are frequently cited as major contributors to security lapses	Both recognize the impact of workload on security practices	Literature rarely addresses this as a standalone factor, while the world highlights it as a significant practical issue
HSV8: Resistance to Change	They are often discussed in a more theoretical context.	They are frequently a barrier to implementing new security measures and protocols.	Both acknowledge that resistance to change can impede security improvements.	The literature discusses it theoretically, while the world focuses on practical resistance issues.
HSV9: Neglecting Secure Software Development Lifecycle	Stresses the importance of incorporating security throughout the development lifecycle	They are often neglected due to cost, time constraints, or lack of immediate perceived benefits.	Both emphasize the necessity of including security in every development lifecycle phase.	Literature focuses on ideal processes, while the world highlights the practical neglect of secure development practices.

 doi:10.20944/preprints202407.0669.v1

HSV10: Cultural Issues	Emphasizes the importance of a security-centric culture within organizations	Struggles to integrate security into core values and daily practices	Both recognize the influence of organizational culture on security practices	Literature provides theoretical and cultural frameworks, while the real world addresses specific cultural challenge
HSV11: Over-Reliance on Tools	Advocates for a balanced approach combining tools with human oversight	Often places excessive reliance on automated tools, underestimating the need for human judgment	Both highlight the risks of relying solely on automated tools for security	Literature focuses on balanced approaches, while the world highlights practical issues due to over-reliance on tools

Table 6 provides a comparative analysis of the impact percentage of human success factors (HSFs) and human security vulnerabilities (HSVs) identified through literature review and empirical study.

Table 6. Comparative Analysis of the Impact Percentage of HSFs and HSVs identified through literature Review and Empirical Survey on SSD Projects.

Human Success Factors (HSFs)	Literature Review Impact %	Empirical Survey Impact %	Human Security Vulnerabilities (HSVs)	Literature Review Impact %	Impact Percentage
HSF1	94	90	HSV1	90	91
HSF2	91	85	HSV2	89	85
HSF3	86	81	HSV3	78	84
HSF4	82	79	HSV4	75	82
HSF5	79	78	HSV5	69	78
HSF6	78	76	HSV6	68	66
HSF7	75	73	HSV7	64	60
HSF8	73	72	HSV8	56	59
HSF9	71	70	HSV9	50	52
HSF10	63	60	HSV10	47	49
HSF11	60	52	HSV11	45	43
HSF12	55	50	HSV12	44	40

Key Observation about Table 6:

- **Top Success Factors:** HSF1 (Skill and Expertise) and HSF2 (Awareness and Mindset) are rated highest in the literature review and empirical survey, indicating their critical importance in SSD projects.
- **Consistency:** Most factors show high consistency between literature and survey impacts, reflecting a general effect on their importance.
- **Lower Impact Factors:** HSF11 (Scalability and Flexibility) and HSF12 (Ethical and Cultural Sensitivity) are rated lower, suggesting these are less emphasized in practical settings than technical and managerial skills.
- **Top Security Vulnerabilities:** HSV1 (Lack of Security Awareness and Training) and HSV2 (Poor Communication and Collaboration) are consistently rated highest, highlighting their significant impact on security.
- **Variability:** There is more variability in the impact percentages of vulnerabilities than success factors, suggesting differences in perceptions of vulnerability impacts.

 doi:10.20944/preprints202407.0669.v1

- **Lower Impact Vulnerabilities:** HSV10 (Cultural Issues), HSV11 (Insufficient Incident Response Preparedness), and HSV12 (Over-Reliance on Tools) are rated lower, indicating these are perceived as less critical compared to others.

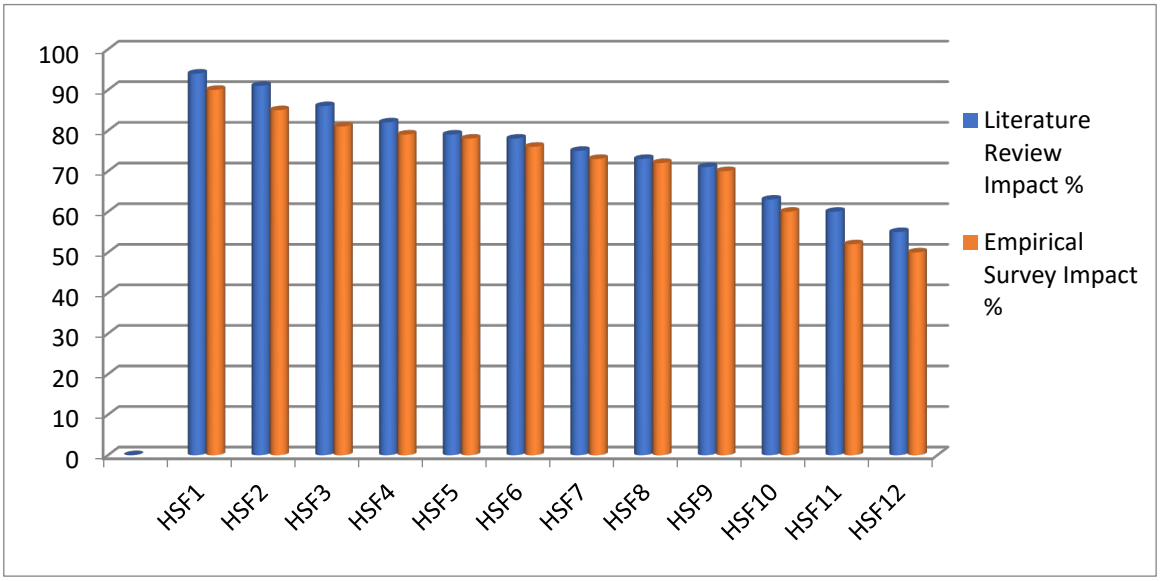

Figure 6. Comparison of Human Success Factors identified through Literature review and Empirical Survey Impact of SSD Projects.

Figure 6 depicts that both the literature review and empirical survey data agree that skills and expertise (HSF1), awareness and mindset (HSF2), and communication and collaboration (HSF3) are critical success factors with very high impact percentages. This indicates a strong consensus on the importance of continuous education and awareness programs to secure the software development processes. Similarly, effective communication and collaboration are crucial for identifying and addressing security issues. Ethical and cultural sensitivity (HSF11), scalability, and flexibility (HSF12) are rated less impactful than other factors.

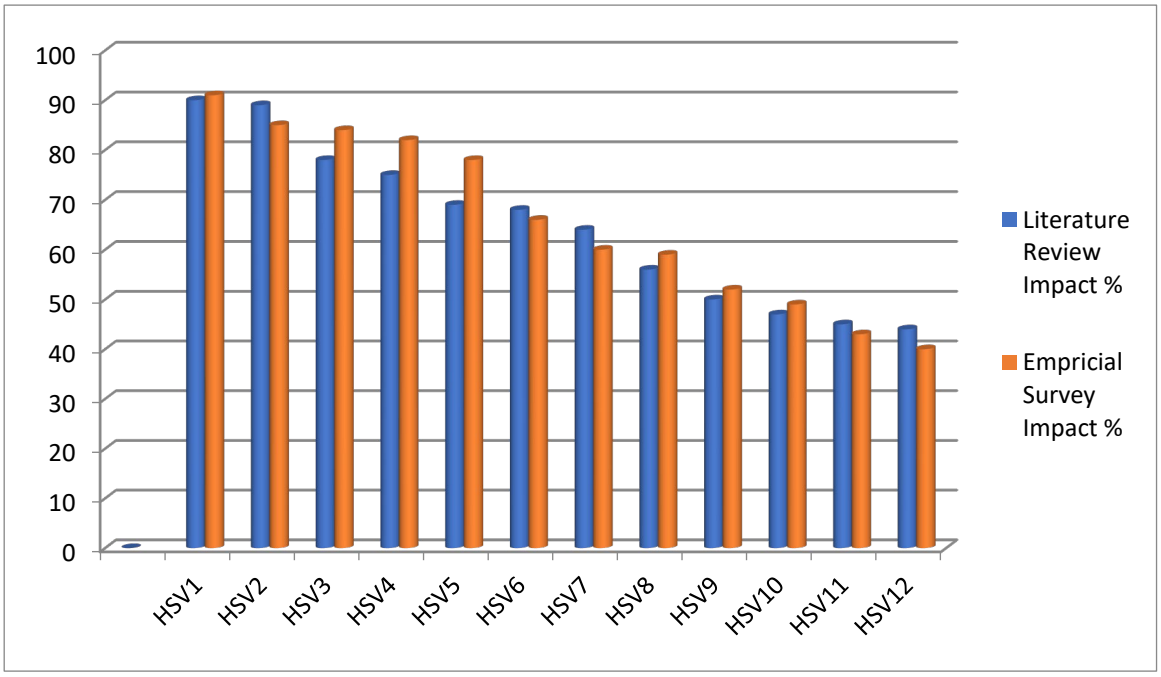

Figure 7. Comparison of Human Security Vulnerabilities identified through Literature review and Empirical Survey Impact of SSD Projects.

 doi:10.20944/preprints202407.0669.v1

Figure 7 shows a high level of agreement between literature reviews and empirical surveys on the impact of various human security vulnerabilities. The percentages are consistently close across most categories. The most critical vulnerabilities identified are lack of awareness and training (HSV1); poor communication and collaboration (HSV2), and inadequate testing and reviews (HSV3). Cultural issues (HSV10) and over-reliance on tools (HSV12) are related as less impactful compared to other factors

4.3. RQ3: What are the Practices For Addressing HSFs and HSVFs, as Identified through Literature Review and Real-World Industries, that Influence SSD Projects?

Addressing human success factors (HSFs) in secure software development (SSD) projects requires a comprehensive approach that balances technical excellence with well-being, collaboration, and continuous improvement of the development team. Table 7 presents some best practices identified through literature review and empirical study.

Table 7. Practices for addressing HSFs that Impact SSD Projects.

Practices for Addressing Human Success Factors	Sub-Practices		
Security Training and Awareness	**Regular Training:** Provide ongoing security training to update the development team on the latest threats, secure coding practices, and regulatory requirements.	**Security Champions:** Identify and empower security champions within the team to advocate for and enforce security best practices.	**Awareness Campaigns:** Run regular security awareness campaigns to reinforce the importance of security in the development lifecycle.
Collaborative Culture	**Team Collaboration:** Foster a culture of collaboration where security is a shared responsibility across all team members, including developers, testers, and operations.	**Cross-Functional Teams:** Encourage forming cross-functional teams that include security experts to integrate security considerations into all stages of development.	**Knowledge Sharing:** Promote knowledge sharing through internal workshops, seminars, and collaborative tools.
Leadership and Management Support	**Executive Support:** Ensure strong support from leadership to prioritize security and allocate necessary resources.	**Clear Vision:** Communicate a clear vision and strategy for integrating security into the software development process.	**Empowerment:** Empower team members to take ownership of security practices and make decisions that enhance security.
Effective Communication	**Open Channels:** Establish open and effective communication channels to discuss security concerns and share updates.	**Regular Meetings:** Hold regular security-focused meetings to review potential vulnerabilities, share best practices, and update the team on new security threats.	**Feedback Mechanisms:** Implement feedback mechanisms to gather input from team members on security practices and improvement areas.
Skill Development	**Continuous Learning:** Encourage continuous learning and professional development in security through certifications, courses, and conferences.	**Mentorship Programs:** Implement mentorship programs where experienced security professionals guide less experienced team members.	**Hands-On Practice:** Provide opportunities for hands-on practice with secure coding, threat modeling, and penetration testing.
User-Centric Security	**User Involvement:** To understand their security concerns and expectations,	**User Feedback:** Collect and act on user feedback regarding security features and usability.	**Usability Testing:** Conduct usability testing focusing on security features to

 doi:10.20944/preprints202407.0669.v1

	involve end-users early in the development process.		ensure they are user-friendly and effective.
Agile and DevSecOps Practices	**Agile Security Integration:** Integrate security practices into Agile workflows to ensure continuous attention to security.	**DevSecOps:** Adopt DevSecOps practices to automate security testing and integrate security into the CI/CD pipeline.	**Security Retrospectives:** Conduct regular security retrospectives to review security incidents, identify lessons learned, and plan improvements.
Clear Roles and Responsibilities	**Defined Roles:** Clearly define roles and responsibilities related to security within the development team.	**Role Flexibility:** Allow role flexibility to adapt to changing security needs and leverage team members' strengths.	**Accountability:** Ensure accountability for security practices and outcomes across all team members.
Stakeholder Engagement	**Stakeholder Communication:** Maintain regular communication with stakeholders to inform them about security measures and risks.	**Expectation Management:** Manage stakeholder expectations by being transparent about security risks and the measures in place to address them.	**Stakeholder Involvement:** Involve stakeholders in security planning and decision-making processes.
Risk Management	**Risk Identification:** Proactively identify potential security risks that could impact the project.	**Mitigation Strategies:** Develop and implement risk mitigation strategies to address identified security risks.	**Incident Response Plans:** Have incident response plans to address security breaches and quickly minimize impact.
Continuous Improvement	**Performance Metrics:** Use security performance metrics to assess security practices' effectiveness and identify areas for improvement.	**Process Optimization:** Continuously optimize security processes based on feedback and performance data.	**Innovation Encouragement:** Encourage innovation in security practices and use new tools and techniques to enhance security.

By implementing Table 7 practices, SSD projects can effectively address HSFs, leading to improved security outcomes, enhanced team performance, and higher stakeholder satisfaction.

Addressing human security vulnerabilities (HSVs) in SSD projects involves training, process improvements, technology, and a strong security culture. Table 8 presents some practices identified through literature review and real-world industries.

Table 8. Practices for addressing HSVs that Impact SSD Projects.

Practices for Addressing Human Security Vulnerabilities	**Sub-Practices**		
Comprehensive Security Training and Awareness	**Regular Training:** Conduct regular training sessions to inform the development team about the latest security threats, secure coding practices, and compliance requirements.	**Phishing Simulations:** Implement phishing simulations to educate employees on recognizing and responding to phishing attacks.	**Security Certifications:** Encourage team members to obtain security certifications such as CISSP, CEH, or CSSLP.
Security Culture and Leadership	**Security Champions:** Designate security champions within development teams to promote security best practices	**Leadership Commitment:** Ensure that leadership demonstrates a commitment to security by prioritizing it in all aspects of the project.	**Reward and Recognition:** Recognize and reward team members who contribute to improving security within the project.

 doi:10.20944/preprints202407.0669.v1

	and act as liaisons with the security team.		
Effective Communication and Collaboration	**Clear Communication Channels:** Establish clear communication channels for reporting security issues and sharing security-related information	**Cross-Functional Teams:** Create cross-functional teams with security experts to integrate security considerations throughout the development lifecycle.	**Regular Security Briefings:** Hold regular security briefings to update the team on new threats and security incidents.
Access Control and Least Privilege	**Role-Based Access Control (RBAC):** Implement RBAC to ensure that team members have access only to the resources they need to perform their jobs.	**Least Privilege Principle:** Apply the principle of least privilege to minimize the potential impact of security breaches,	**Multi-Factor Authentication (MFA):** Use MFA to add extra security to critical systems and applications.
Secure Development Practices	**Secure Coding Standards:** Adopt and enforce secure coding standards (e.g., OWASP Secure Coding Practices) to prevent common vulnerabilities.	**Code Review:** Regular code reviews should focus on security to identify and address vulnerabilities early.	**Static and Dynamic Analysis:** Use static and dynamic analysis tools to detect security issues in the code.
Incident Response and Management	**Incident Response Plan:** Develop and maintain an incident response plan to quickly and effectively address security incidents.	**Simulation Drills:** Conduct regular incident response drills to ensure the team is prepared to handle security breaches.	**Post-Incident Analysis:** Perform post-incident analysis to learn from security incidents and improve future responses.
Continuous Monitoring and Testing	**Vulnerability Scanning:** Implement regular vulnerability scanning to identify and address security weaknesses.	**Penetration Testing:** Conduct penetration testing to simulate attacks and evaluate the effectiveness of security measures.	**Security Audits:** Perform regular security audits to ensure compliance with security policies and standards.
Data Protection and Privacy	**Encryption:** Use encryption to protect sensitive data both at rest and in transit.	**Data Minimization:** Collect and retain only the data necessary for the project to reduce the risk of data breaches.	**Privacy by Design:** Incorporate privacy considerations into the design and development of software to protect user data.
Supply Chain Security	**Third-Party Risk Management:** Assess and manage the security risks associated with third-party vendors and partners.	**Secure Supply Chain:** Implement security measures to protect the software supply chain, including code signing and verification.	**Contractual Security Requirement:** Include security requirements in contracts with third-party vendors to ensure they adhere to security best practices.
User Education and Support	**Security Awareness for Users:** Educate users about security best practices and how to recognize potential threats.	**User-Friendly Security Features:** Design security features that are user-friendly and encourage proper security behaviors.	**Support Channels:** Provide clear support channels for users to report security concerns and get help with security-related issues.
Regulatory Compliance and Standards	**Compliance Programs:** Establish compliance programs to adhere to relevant security regulations and standards (e.g., GDPR, HIPAA).	**Regular Audits:** Conduct regular audits to ensure ongoing compliance with security regulations and standards.	**Documentation:** Maintain thorough documentation of security policies, procedures, and compliance efforts.

 doi:10.20944/preprints202407.0669.v1

By implementing Table 8 practices, SSD projects can effectively address HSVs, leading to more resilient and improved security outcomes.

4.4. RQ4: How Can a Decision-Making Framework Be Developed to Identify the Weights of Maintenance and Deployment Security Risks in Healthcare System Software Development?

In this paper, we employed the coding system of Strauss' [49] ground theory (GT) technique to identify, classify, and organize the identified practices for HSFs and HSVs that impact SSD projects. Although we have already collected the data through a literature review, we used the four main phases of the GT coding scheme to map the practices into four major categories (i.e., "code," "categories", "sub-categories," and theory/theoretical model"). All the authors of this study are on the mapping team. First, we allocated a unique code/label to each practice we studied. The second phase involved categorizing the examined practices into nine broad phases/groups, “C1: Security Training and Awareness”, “C2: Collaborative Security Culture and Leadership”, “C3: Effective Communication and Collaboration”, “C4: Skill Development and Stakeholder Engagement”, “C5: Agile and DevSecOps Practices”, “C6: User-Centric Security, Clear Roles and Responsibilities”, “C7: Supply Chain Security and Risk Management”, “C8: Continuous Improvement, Monitoring and Testing”, and “C9: Data Protection and Privacy, Access Control and Least Privilege”. In the third step, the sub-practices were mapped into these categories. In the fourth step, we engineered a theoretical model depicted in Figure 8.

 doi:10.20944/preprints202407.0669.v1

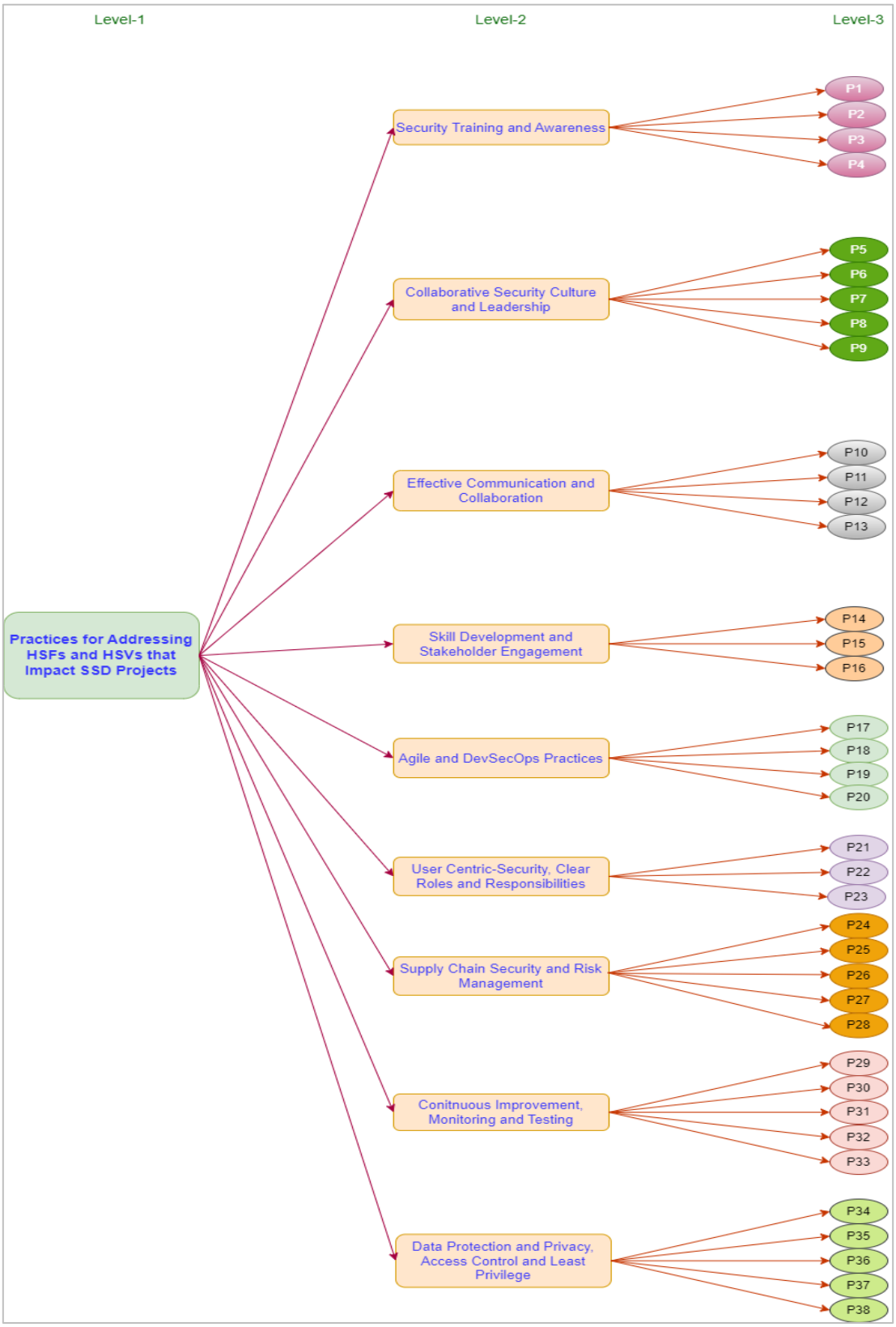

Figure 8. Theoretical Model of the Practices for HSFs and HSVs that impact SSD Projects.

The primary goal of this categorization is to construct a hierarchical framework for executing the FAHP. Furthermore, this categorization will help academic researchers and practitioners to identify the most essential practice for HSFs and HSVs in the context of SSD. We identified 38 practices mapped in each category, as stated in Table 9.

 doi:10.20944/preprints202407.0669.v1

Table 9. HSFs and HSVs Practices Categories and Subcategories.

HSFs and HSVs: Practices Categories	HSFs and HSVs: Practices Subcategories
C1: Security Training and Awareness	P1: Regular Training and Phishing Simulations P2: Security and Awareness Campaigns P3: Security Certifications and Support Channels P4: User-Friendly Security Features
C2: Collaborative Security Culture and Leadership	P5: Team Collaboration and Cross-Functional Teams P6: Knowledge Sharing and Executive Support P7: Leadership Commitment P8: Reward and Recognition P9: Clear Vision and Empowerment
C3: Effective Communication and Collaboration	P10: Open Channels and Regular Meetings P11: Feedback Mechanisms P12: Clear Communication Channels P13: Regular Security Briefings
C4: Skill Development and Stakeholder Engagement	P14: Continuous Learning and Mentorship Programs P15: Hands-On Practice and Stakeholder Communication P16: Expectation Management and Stakeholder Involvement
C5: Agile and DevSecOps Practices	P17: Agile Security Integration and DevSecOps P18: Security Retrospectives P19: Secure Coding Standards P20: Code Review, Static and Dynamic Analysis
C6: User-Centric Security, Clear Roles and Responsibilities	P21: Defined Roles and Role Flexibility P22: Accountability and Usability Testing P23: User Involvement and User Feedback
C7: Supply Chain Security and Risk Management	P24: Risk Identification and Mitigation Strategies P25: Incident Response Plans and Post-Incident Analysis P26: Simulation Drills P27: Third-Party Risk Management P28: Secure Supply Chain and Contractual Security
C8: Continuous Improvement, Monitoring and Testing	P29: Performance Metrics and Process Optimization P30: Innovation Encouragement and Compliance Programs P31: Vulnerability Scanning and Penetration Testing P32: Security and Regular Audits P33: Documentation
C9: Data Protection and Privacy, Access Control and Least Privilege	P34: Role-Based Access Control (RBAC) P35: Least Privilege Principle P36: Multi-Factor Authentication (MFA) P37: Encryption and Data Minimization P38: Privacy by Design

This section presents an analysis of the results obtained from the fuzzy AHP. The FAHP steps were meticulously executed to ascertain HSF and HSV practice weights for SSD projects within and across categories. Subsequent subsections detail the complete sequence of FAHP steps undertaken, along with their corresponding implications:

4.4.1. Step-1: Sorting out a problem's components (HSFs and HSVs practices) using a hierarchical framework

Motivated by existing literature [45, 50-52], we constructed a hierarchical structure to address progressively complex decision-making problems. This hierarchy consisted of three levels, as depicted in Figure 8.

 doi:10.20944/preprints202407.0669.v1

At Level 1, the primary issue (prioritization of HSFs and HSVs for SSD projects) was outlined. Level 2 encompassed the categories of practices, while Level 3 detailed their respective practices, as illustrated in Figure 8.

4.4.2. Step-2: Comparing Matrix Pairs

In Step 2, within the FAHP analysis, our objective was to rank the identified HSF and HSV practices based on their significance within SSD projects. This required a pair-wise matrix comparison and expert input to prioritize practices and categories. We expanded upon our first survey with a Fuzzy-AHP version, but only 25 of 70 people who were invited to take part were willing to do so. Rigorous checks were applied to ensure consistency and completeness of the 20 complete responses obtained from healthcare security experts for the pair-wise matrix comparison. While the sample size 20 might seem limited, it aligns with comparable studies [45, 50-53] and allows for reasonable generalization. The geometric mean formula below-converted expert survey responses to Triangular Fuzzy Numbers (TFN):

$$\text{Geometric mean} = \sqrt[n]{a1 \times a2 \times a3 \ldots \ldots an}$$

a = weight of each response

n = number of responses

Table 10. Conversion scale of Triangular Fuzzy numbers [54]

Linguistic Scale	Triangular Fuzzy Scale	Triangular Fuzzy Reciprocal Scale
Just Equal (JE)	(1, 1, 1)	(1, 1, 1)
Equally Important (EI)	(0.5, 1, 1.5)	(0.6, 1, 2)
Weakly Important (WI)	(1, 1.5, 2)	(0.5, 0.6, 1)
Strongly More Important (SMI)	(1.5, 2, 2.5)	(0.4, 0.5, 0.6)
Very Strongly More Important (VSMI)	(2, 2.5, 3)	(0.3, 0.4, 0.5)
Absolutely More Important (AMI)	(2.5, 3, 3.5)	(0.2, 0.3, 0.4)

.

Table 10 shows the fuzzy triangle values used to apply the linguistic variable. As described in [54], the triangular matrix method was used to formulate pair-wise comparison matrices. Table 11 displays the paired matrix comparison of HSF and HSV practice categories for SSD projects.

Table 11. Fuzzified Pair-wise Matrix comparison of the Practices Categories for HSFs and HSVs.

Categories	C-1	C-2	C-3	C-4	C-5	C-6	C-7	C-8	C-9	Sum	Sum * Inverse of the total sum of the columns
C-1	(1, 1, 1)	(0.4, 0.5, 0.6)	(0.5, 1, 1.5)	(0.5, 1, 1.5)	(0.5, 0.6, 1)	(0.5, 1, 1.5)	(0.6, 1, 2)	(0.6, 1, 2)	(0.5, 0.6, 1)	(5.1, 7.7, 12.1)	(0.0392, 0.0893, 0.2008)
C-2	(1.5, 2, 2.5)	(1, 1, 1)	(0.5, 1, 1.5)	(0.4, 0.5, 0.6)	(0.5, 1, 1.5)	(1, 1.5, 2)	(0.5, 0.6, 1)	(0.5, 0.6, 1)	(1, 1.5, 2)	(7.4, 9.7, 13.1)	(0.0569, 0.1125, 0.2174)
C-3	(0.6, 1, 2)	(0.6, 1, 2)	(1, 1, 1)	(1, 1.5, 2)	(1.5, 2, 2.5)	(0.5, 1, 1.5)	(0.5, 1, 1.5)	(1.5, 2, 2.5)	(0.6, 1, 2)	(7.8, 11.5, 17)	(0.0600, 0.1334, 0.2822)
C-4	(0.6, 1, 2)	(1.5, 2, 2.5)	(0.5, 0.6, 1)	(1, 1, 1)	(0.5, 1, 1.5)	(1.5, 2, 2.5)	(0.5, 1, 1.5)	(0.5, 1, 1.5)	(0.5, 1, 1.5)	(7.1, 10.6, 15)	(0.0546, 0.1229, 0.2490)
C-5	(1, 1.5, 2)	(0.6, 1, 2)	(0.4, 0.5, 0.6)	(0.6, 1, 2)	(1, 1, 1)	(0.6, 1, 2)	(0.5, 0.6, 1)	(0.6, 1, 2)	(0.4, 0.5, 0.6)	(5.7, 8.1, 16.2)	(0.0438, 0.0939, 0.2689)
C-6	(0.6, 1, 2)	(0.5, 0.6, 1)	(0.6, 1, 2)	(0.4, 0.5, 0.6)	(0.5, 1, 1.5)	(1, 1, 1)	(1, 1.5, 2)	(1, 1.5, 2)	(1.5, 2, 2.5)	(7.1, 10.1, 14.6)	(0.0546, 0.1171, 0.2423)
C-7	(0.5, 1, 1.5)	(1, 1.5, 2)	(0.6, 1, 2)	(0.6, 1, 2)	(1, 1.5, 2)	(0.5, 0.6, 1)	(1, 1, 1)	(0.5, 0.6, 1)	(0.5, 0.6, 1)	(6.2, 8.8, 13.5)	(0.0477, 0.1020, 0.2241)

 doi:10.20944/preprints202407.0669.v1

C-8	(0.5, 1, 1.5)	(1, 1.5, 2)	(0.4, 0.5, 0.6)	(0.6, 1, 2)	(0.5, 1, 1.5)	(0.5, 0.6, 1)	(1, 1.5, 2)	(1, 1, 1)	(1, 1.5, 2)	(6.5, 9.6, 13.6)	(0.0500, 0.1113, 0.2257)
C-9	(1, 1.5, 2)	(0.5, 0.6, 1)	(0.5, 1, 1.5)	(0.6, 1, 2)	(1.5, 2, 2.5)	(0.4, 0.5, 0.6)	(1, 1.5, 2)	(0.5, 0.6, 1)	(1, 1, 1)	(7, 9.7, 13.6)	(0.0539, 0.1125, 0.2257)
The total sum of the columns										(59.9, 85.8, 128.7)	
The inverse of the total sum of the columns										(0.0077, 0.0116, 0.0166)	

4.4.3. Step 3: Consistency Evaluation

The practices Categories matrix (refer to Table 12) was employed to assess consistency. This table illustrates the Fuzzy Crisp Matrix (FCM) derived from de-fuzzifying practices. It divides categories into crisp numbers through pair-wise matrix comparisons using equation (14).

Table 12. Fuzzy Crisp Matrix of Practices Categories for HSFs and HSVs that impact SSD Projects.

Categories	C-1	C-2	C-3	C-4	C-5	C-6	C-7	C-8	C-9
C-1	1	0.5	1	1	0.61	1	1	1	0.61
C-2	2	1	1	0.5	1	1.5	0.61	0.61	1.5
C-3	1	1	1	1.5	2	1	1	2	0.61
C-4	1	2	0.61	1	1	2	1	1	1
C-5	1.5	1	0.5	1	1	1	0.61	1	0.5
C-6	1	0.61	1	0.5	1	1	1.5	1.5	2
C-7	1	1.5	1	1	1.5	0.61	1	0.61	0.61
C-8	1	1.5	0.5	1	1	0.61	1.5	1	1.5
C-9	1.5	0.61	1	1	2	0.5	1.5	0.61	1
Sum	11	9.72	7.61	8.5	11.11	9.2	9.72	9.33	9.32

4.4.4. Step 4: Identification of Local Priority Weight of HSFs and HSVs Practices Categories

a. A Numerical Example

To find the weight vector's normalized values, we used Equation (15). Tables 13 and 14 show the priority weights assigned to each HSF and HSV practice category in the development of the SSD project, and these values are normalized and non-fuzzy.

Table 13. Degree of Possibilities of Practice Categories for HSFs and HSVs.

Categories	Sum * Inverse of the total sum of the columns	C-1	C-2	C-3	C-4	C-5	C-6	C-7	C-8	C-9	Priority Weight
V (C1≥…)	(0.0392, 0.0893, 0.2008) (l1, m1, u1)	-	0.6625	0.5991	0.6359	0.7416	0.6523	0.7091	0.6765	0.6671	0.5991
V (C2≥…)	(0.0569, 0.1125, 0.2174) (l2, m2, u2)	1	-	0.8827	0.9399	1	0.9725	1	1	1	0.8827
V (C3≥…)	(0.0600, 0.1334, 0.2822) (l3, m3, u3)	1	1	-	1	1	1	1	1	1	1
V (C4≥…)	(0.0546, 0.1229, 0.2490) (l4, m4, u4)	1	1	0.9473	-	1	1	1	1	1	0.9473

 doi:10.20944/preprints202407.0669.v1

V (C5≥…)	(0.0438, 0.0939, 0.2689) (l5, m5, u5)	1	0.9193	0.8409	0.8808	-	0.9023	0.9646	0.9263	0.9203	0.8409
V (C6≥…)	(0.0546, 0.1171, 0.2423) (l6, m6, u6)	1	1	0.9179	0.9700	1	-	1	1	1	0.9179
V (C7≥…)	(0.0477, 0.1020, 0.2241) (l7, m7, u7)	1	0.9409	0.8393	0.8902	1	0.9182	-	0.9492	0.9418	0.8393
V (C8≥…)	(0.0500, 0.1113, 0.2257) (l8, m8, u8)	1	0.9929	0.8823	0.9365	1	0.9672	1	-	0.9930	0.8823
V (C9≥…)	(0.0539, 0.1125, 0.2257) (l9, m9, u9)	1	1	0.8879	0.9427	1	0.9738	1	1	-	0.8879

Table 14. Normalized Matrix of Practices Categories for HSFs and HSVs.

Categories	C-1	C-2	C-3	C-4	C-5	C-6	C-7	C-8	C-9	W
C-1	0.09090	0.05144	0.13140	0.11764	0.05490	0.10869	0.10288	0.10718	0.06545	0.09227
C-2	0.18181	0.10288	0.13140	0.05882	0.09000	0.16304	0.06275	0.06538	0.16094	0.11300
C-3	0.09090	0.10288	0.13140	0.05882	0.18001	0.10869	0.10288	0.21436	0.06545	0.11726
C-4	0.09090	0.20576	0.08015	0.11764	0.09000	0.21739	0.10288	0.10718	0.10729	0.12435
C-5	0.13636	0.10288	0.06570	0.11764	0.09000	0.10869	0.06275	0.10718	0.05364	0.09387
C-6	0.09090	0.06275	0.13140	0.05882	0.09000	0.10869	0.06275	0.16077	0.21459	0.10896
C-7	0.09090	0.15432	0.13140	0.11764	0.13501	0.06630	0.10288	0.06538	0.06545	0.10325
C-8	0.09090	0.15432	0.06570	0.11764	0.09000	0.06630	0.15432	0.10718	0.16094	0.11922
C-9	0.13636	0.06275	0.13140	0.11764	0.18001	0.05434	0.15432	0.06538	0.10729	0.11216

$$\lambda_{max} = (11 \times 0.09227) + (9.72 \times 0.11300) + (7.61 \times 0.11726) + (8.5 \times 0.12435) + (11.11 \times 0.09387) + (9.2 \times 0.10896) + (9.72 \times 0.10325) + (9.33 \times 0.11922) + (9.32 \times 0.11216)$$

$$= 1.01431 + 1.09836 + 0.89234 + 1.05697 + 1.04289 + 1.00243 + 1.00359 + 1.11232 + 1.04533$$

$$= 9.26854$$

In Table 2, the random index (RI) is determined as 1.45, considering the presence of 9 elements. Equations (15) and (16) were used to calculate the consistency index (CI) and, by extension, the consistency ratio (CR):

$$CI = \frac{(9.26854 - 9)}{9-1}$$

$$CI = \frac{0.26854}{8}$$

$$CI = 0.03356$$

$$CR = \frac{CI}{RI}$$

$$CR = \frac{0.03356}{1.45}$$

$$CR = 0.02314$$

CR value of 0.02314, which is <0.10

This CR value of 0.02314, being <0.10, indicates the consistency of the pair-wise matrix representing HSF and HSV practice categories.

 doi:10.20944/preprints202407.0669.v1

Tables 15–24 show the results of applying these same procedures to assign weights to each HSF and HSV practice category, its practices, and the fuzzified pair-wise matrix comparison.

Table 15. Comparison of Fuzzified Pair-wise Matrix for C-1: Security Training and Awareness.

C-1	P-1	P-2	P-3	P-4	Weight
P-1	(1, 1, 1)	(0.5, 1, 1.5)	(0.5, 0.6, 1)	(1, 1.5, 2)	0.2478
P-2	(0.6, 1, 2)	(1, 1, 1)	(0.5, 1, 1.5)	(0.5, 1, 1.5)	0.2447
P-3	(1, 1.5, 2)	(0.6, 1, 2)	(1, 1, 1)	(0.5, 0.6, 1)	0.2531
P-4	(0.5, 0.6, 1)	(0.6, 1, 2)	(1, 1.5, 2)	(1, 1, 1)	0.2543

Table 16: Comparison of Fuzzified Pair-wise Matrix for C-2: Collaborative Security Culture and Leadership.

C-2	P-5	P-6	P-7	P-8	P-9	Weight
P-5	(1, 1, 1)	(0.5, 0.6, 1)	(2, 2.5, 3)	(0.5, 0.6, 1)	(2.5, 3, 3.5)	0.2727
P-6	(1, 1.5, 2)	(1, 1, 1)	(1, 1.5, 2)	(0.5, 1, 1.5)	(0.4, 0.5, 0.6)	0.1764
P-7	(0.3, 0.4, 0.5)	(0.5, 0.6, 1)	(1, 1, 1)	(2.5, 3, 3.5)	(0.5, 1, 1.5)	0.1985
P-8	(1, 1.5, 2)	(0.6, 1, 2)	(0.2, 0.3, 0.4)	(1, 1, 1)	(0.3, 0.4, 0.5)	0.1134
P-9	(0.2, 0.3, 0.4)	(1.5, 2, 2.5)	(0.6, 1, 2)	(2, 2.5, 3)	(1, 1, 1)	0.2388

Table 17. Comparison of Fuzzified Pair-wise Matrix for C-3: Effective Communication and Collaboration.

C-3	P-10	P-11	P-12	P-13	Weight
P-10	(1, 1, 1)	(0.2, 0.3, 0.4)	(1, 1.5, 2)	(1.5, 2, 2.5)	0.2588
P-11	(2.5, 3, 3.5)	(1, 1, 1)	(0.4, 0.5, 0.6)	(0.5, 1, 1.5)	0.3038
P-12	(0.5, 0.6, 1)	(1.5, 2, 2.5)	(1, 1, 1)	(0.5, 0.6, 1)	0.2219
P-13	(0.4, 0.5, 0.6)	(0.6, 1, 2)	(1, 1.5, 2)	(1, 1, 1)	0.2153

Table 18. Comparison of Fuzzified Pair-wise Matrix for C4: Skill Development and Stakeholder Engagement.

C-4	P-14	P-15	P-16	Weight
P-14	(1, 1, 1)	(0.4, 0.5, 0.6)	(0.5, 1, 1.5)	0.2554
P-15	(1.5, 2, 2.5)	(1, 1, 1)	(0.6, 1, 2)	0.4142
P-16	(0.6, 1, 2)	(0.5, 1, 1.5)	(1, 1, 1)	0.3302

Table 19. Comparison of Fuzzified Pair-wise Matrix for C5: Agile and DevSecOps Practices.

C-5	P-17	P-18	P-19	P-20	Weight
P-17	(1, 1, 1)	(1.5, 2, 2.5)	(0.5, 1, 1.5)	(0.5, 1, 1.5)	0.2902
P-18	(0.4, 0.5, 0.6)	(1, 1, 1)	(0.5, 1, 1.5)	(1.5, 2, 2.5)	0.2475
P-19	(0.6, 1, 2)	(0.6, 1, 2)	(1, 1, 1)	(1, 1.5, 2)	0.2707
P-20	(0.6, 1, 2)	(0.4, 0.5, 0.6)	(0.5, 0.6, 1)	(1, 1, 1)	0.1914

Table 20. Comparison of Fuzzified Pair-wise Matrix for C6: User-Centric Security, Clear Roles and Responsibilities.

C-6	P-21	P-22	P-23	Weight
P-21	(1, 1, 1)	(0.4, 0.5, 0.6)	(1, 1.5, 2)	0.2868
P-22	(1.5, 2, 2.5)	(1, 1, 1)	(0.6, 1, 2)	0.3978
P-23	(0.5, 0.6, 1)	(0.5, 1, 1.5)	(1, 1, 1)	0.3153

 doi:10.20944/preprints202407.0669.v1

Table 21. Comparison of Fuzzified Pair-wise Matrix for C7: Supply Chain Security and Risk Management.

C-7	P-24	P-25	P-26	P-27	P-28	Weight
P-24	(1, 1, 1)	(0.5, 1, 1.5)	(0.6, 1, 2)	(1, 1.5, 2)	(0.5, 0.6, 1)	0.1964
P-25	(0.6, 1, 2)	(1, 1, 1)	(1.5, 2, 2.5)	(0.4, 0.5, 0.6)	(2, 2.5, 3)	0.2373
P-26	(0.5, 1, 1.5)	(0.4, 0.5, 0.6)	(1, 1, 1)	(0.3, 0.4, 0.5)	(0.5, 1, 1.5)	0.1346
P-27	(0.5, 0.6, 1)	(1.5, 2, 2.5)	(2, 2.5, 3)	(1, 1, 1)	(0.6, 1, 2)	0.2519
P-28	(1, 1.5, 2)	(0.3, 0.4, 0.5)	(0.6, 1, 2)	(0.5, 1, 1.5)	(1, 1, 1)	0.1794

Table 22. Comparison of Fuzzified Pair-wise Matrix for C8: Continuous Improvement, Monitoring and Testing.

C-8	P-29	P-30	P-31	P-32	P-33	Weight
P-29	(1, 1, 1)	(1.5, 2, 2.5)	(0.5, 1, 1.5)	(0.5, 1, 1.5)	(1, 1.5, 2)	0.2379
P-30	(0.4, 0.5, 0.6)	(1, 1, 1)	(0.5, 1, 1.5)	(1.5, 2, 2.5)	(0.5, 1, 1.5)	0.1887
P-31	(0.6, 1, 2)	(0.6, 1, 2)	(1, 1, 1)	(1, 1.5, 2)	(1.5, 2, 2.5)	0.2376
P-32	(0.6, 1, 2)	(0.4, 0.5, 0.6)	(0.5, 0.6, 1)	(1, 1, 1)	(0.5, 1, 1.5)	0.1652
P-33	(0.5, 0.6, 1)	(0.6, 1, 2)	(0.4, 0.5, 0.6)	(0.6, 1, 2)	(1, 1, 1)	0.1701

Table 23. Comparison of Fuzzified Pair-wise Matrix for C9: Data Protection and Privacy, Access Control and Least Privilege.

C-9	P-34	P-35	P-36	P-37	P-38	Weight
P-34	(1, 1, 1)	(2, 2.5, 3)	(0.4, 0.5, 0.6)	(0.6, 1, 2)	(1, 1.5, 2)	0.2308
P-35	(0.3, 0.4, 0.5)	(1, 1, 1)	(2, 2.5, 3)	(1, 1.5, 2)	(0.4, 0.5, 0.6)	0.1767
P-36	(1.5, 2, 2.5)	(0.3, 0.4, 0.5)	(1, 1, 1)	(2.5, 3, 3.5)	(0.5, 1, 1.5)	0.2633
P-37	(0.5, 1, 1.5)	(0.5, 0.6, 1)	(0.2, 0.3, 0.4)	(1, 1, 1)	(1.5, 2, 2.5)	0.1549
P-38	(0.5, 0.6, 1)	(1.5, 2, 2.5)	(0.6, 1, 2)	(0.4, 0.5, 0.6)	(1, 1, 1)	0.1739

4.4.5. Step 5: How to calculate the local and global weights and ranks of HSF and HSV practices

The significance of various categories and HSF and HSV practices in SSD projects was assessed through a weighted analysis. Table 24 displays the rankings of these categories and their practices, calculated using local and global weights. Local weights were assigned to rank each category alongside its associated risk, indicating the relative importance of practice within its category based on its position in the local ranking. Global weights were then computed for each practice to establish their global ranking. These global weights were derived by multiplying the local weight of a practice by the respective category weights. For instance, the category "C4: Skill Development and Stakeholder Engagement" is ranked highest at rank-1 and possesses the greatest weight of 0.12435, as detailed in Table 24, among the identified HSFs and HSVs practices for SSD projects. Similarly, Table 24 shows the highest global weight, which is 0.051506, and the global ranked (rank-1) HSF and HSV practice is "P15: Hands-On Practice and Stakeholder Communication".

Table 24. Fuzzified Local and Global Weights of Practices and Sub-Practices Categories for Addressing HSFs and HSVs that Impact SSD Project.

Practices Categories for Addressing HSFs and HSVs that Impact SSD Project	Categories Weights	Sub-Practices	Local Weight	Local Rank	Global Weight	Global Rank
C1: Security Training and Awareness	0.09227	P1: Regular Training and Phishing Simulations	0.2478	3	0.022865	24

 doi:10.20944/preprints202407.0669.v1

		P2: Security and Awareness Campaigns	0.2447	4	0.022578	25
		P3: Security Certifications and Support Channels	0.2531	2	0.023354	22
		P4: User-Friendly Security Features	0.2543	1	0.023464	21
C2: Collaborative Security Culture and Leadership	0.11300	P5: Team Collaboration and Cross-Functional Teams	0.2727	2	0.030815	8
		P6: Knowledge Sharing and Executive Support	0.1764	4	0.019933	30
		P7: Leadership Commitment	0.1985	3	0.022431	27
		P8: Reward and Recognition	0.1134	5	0.012814	38
		P9: Clear Vision and Empowerment	0.2388	1	0.026984	14
C3: Effective Communication and Collaboration	0.11726	P10: Open Channels and Regular Meetings	0.2588	2	0.030347	9
		P11: Feedback Mechanisms	0.3038	1	0.035624	4
		P12: Clear Communication Channels	0.2219	3	0.02602	15
		P13: Regular Security Briefings	0.2153	4	0.025246	19
C4: Skill Development and Stakeholder Engagement	**0.12435**	P14: Continuous Learning and Mentorship Programs	0.2554	3	0.031759	6
		P15: Hands-On Practice and Stakeholder Communication	**0.4142**	**1**	**0.051506**	**1**
		P16: Expectation Management and Stakeholder Involvement	0.3302	2	0.04106	3
C5: Agile and DevSecOps Practices	0.09387	P17: Agile Security Integration and DevSecOps	0.2902	1	0.027241	13
		P18: Security Retrospectives	0.2475	3	0.023233	23
		P19: Secure Coding Standards	0.2707	2	0.025411	18
		P20: Code Review, Static and Dynamic Analysis	0.1914	4	0.017967	35
C6: User-Centric Security, Clear Roles and Responsibilities	0.10896	P21: Defined Roles and Role Flexibility	0.2868	3	0.03125	7
		P22: Accountability and Usability Testing	0.3978	1	0.043344	2
		P23: User Involvement and User Feedback	0.3153	2	0.034355	5
C7: Supply Chain Security and Risk Management	0.10325	P24: Risk Identification and Mitigation Strategies	0.1964	3	0.020278	29
		P25: Incident Response Plans and Post-Incident Analysis	0.2373	2	0.024501	20
		P26: Simulation Drills	0.1346	5	0.013897	38
		P27: Third-Party Risk Management	0.2519	1	0.026009	16
		P28: Secure Supply Chain and Contractual Security	0.1794	4	0.018523	34

 doi:10.20944/preprints202407.0669.v1

C8: Continuous Improvement, Monitoring and Testing	0.11922	P29: Performance Metrics and Process Optimization	0.2379	1	0.028362	11
		P30: Innovation Encouragement and Compliance Programs	0.1887	3	0.022497	26
		P31: Vulnerability Scanning and Penetration Testing	0.2376	2	0.028327	12
		P32: Security and Regular Audits	0.1652	5	0.019695	32
		P33: Documentation	0.1701	4	0.020279	28
C9: Data Protection and Privacy, Access Control and Least Privilege	0.11216	P34: Role-Based Access Control (RBAC)	0.2308	2	0.025887	17
		P35: Least Privilege Principle	0.1767	3	0.019819	31
		P36: Multi-Factor Authentication (MFA)	0.2633	1	0.029532	10
		P37: Encryption and Data Minimization	0.1549	5	0.017374	36
		P38: Privacy by Design	0.1739	4	0.019505	33

5. Securing Software Development through People Maturity: A Fuzzy-AHP Decision-Making Framework

To illustrate the impact of each human success factor (HSF) and human security vulnerability (HSV) practice for secure software development (SSD), we have constructed a prioritization-oriented taxonomy depicted in Figure 8. This taxonomy outlines the relative priority of each practice within its category. It compares them across all categories in securing software development through people maturity. Visualizing both local and global impacts of each practice aids practitioners in discerning the most significant practice relevant to their roles and responsibilities. The insights in Figure 8 are intended to guide the software development team maturing in refining and formulating new strategies to execute SSD projects. This prioritization-oriented taxonomy serves as a focal point, enabling the team to concentrate on critical areas essential for the success of the SSD projects within the software engineering landscape.

In Figure 8, the category "C4: Skill Development and Stakeholder Engagement" is ranked highest at rank-1 and possesses the most significant weight of 0.12435, as detailed in Table 19, among the identified HSFs and HSVs practices for SSD projects. 'Skill Development and Stakeholder Engagement' remain core prerequisites for SSD because they are the primary determinants of overall security and project success [55]. A Skilled team ensures that the personnel involved have the skills and the necessary and updated knowledge of how to integrate good security standards right from the development process. This learning never stops the organization from checking and eliminating possible risks, using recommended standards, and maintaining relevance with threats. On the other hand, stakeholder engagement specifically refers to embracing every stakeholder during the project implementation, coordinating with the clients, users, or any other interested party. Since people are encouraged to communicate and create synergy, it guarantees that security concerns are clearly defined, valued, and implemented in the process. Stakeholders are involved in offering feedback, contributing to the initiatives regarding the implementation of security policies, and assisting in the direction in which specific objectives of the project should be aligned with those of the organization. Captured jointly, skill enhancement and stakeholder participation ensure that everyone within the software supply chain comprehends security and works towards hardening the output to create adequately secure, dependable, and efficient software solutions.

Similarly, in Figure 8, the highest global weight is 0.051506, and the global ranked (rank-1) HSF and HSV practice is "P15: Hands-On Practice and Stakeholder Communication" [56]. Action and engagement with the projects' stakeholders can be identified as key components in guaranteeing success and security in software development projects. Gaining experience implies interactivity with

 doi:10.20944/preprints202407.0669.v1

the tools, technologies, and methodologies used to develop secure software. A direct interaction of this nature improves their working knowledge and skills, thus making them aware and better placed to avoid security risks. It also helps to develop a culture of minimized stoppages and constant learning, which is especially important in cybersecurity due to its high dynamics.

On the other hand, the stakeholders' communication entails all the involved parties, such as the developers, project managers, clients, and the project's end-users, to be in Compliance with the project's goals, requirements, and limitations. Thus, communication facilitates a clear description of expectations for security, an assessment of the consequences of failed security, and the evaluation of the need for security and the degree to which essential measures are taken. It also encourages timely feedback, decision-making, and teamwork, allowing for the quick identification of new risks and developing ways of handling them. Thus, effectively strengthening practical orientations in developing software solutions and improving communication with key stakeholders can create more secure, intrinsically reliable, and friendly end-user applications.

The taxonomy also presents the variation in the impact of identified HSF and HSV practices concerning their position in their respective categories and overall people's maturity towards SSD projects. For example, the "C1: Security Training and Awareness" category comprises four practices. According to the local ranking in Figure 8, "P4: User-Friendly Security Features" is ranked as the highest priority practice in C-1, but considering the global ranking, it stands at 21st. Similarly, P3 and P1 stand at 2nd and 3rd in C-1, but both stand at 22nd and 24th concerning global ranking. Hence, the SSD team needs to consider both ranking orders (local and global) while developing secure software projects.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 8 July 2024 doi:10.20944/preprints202407.0669.v1

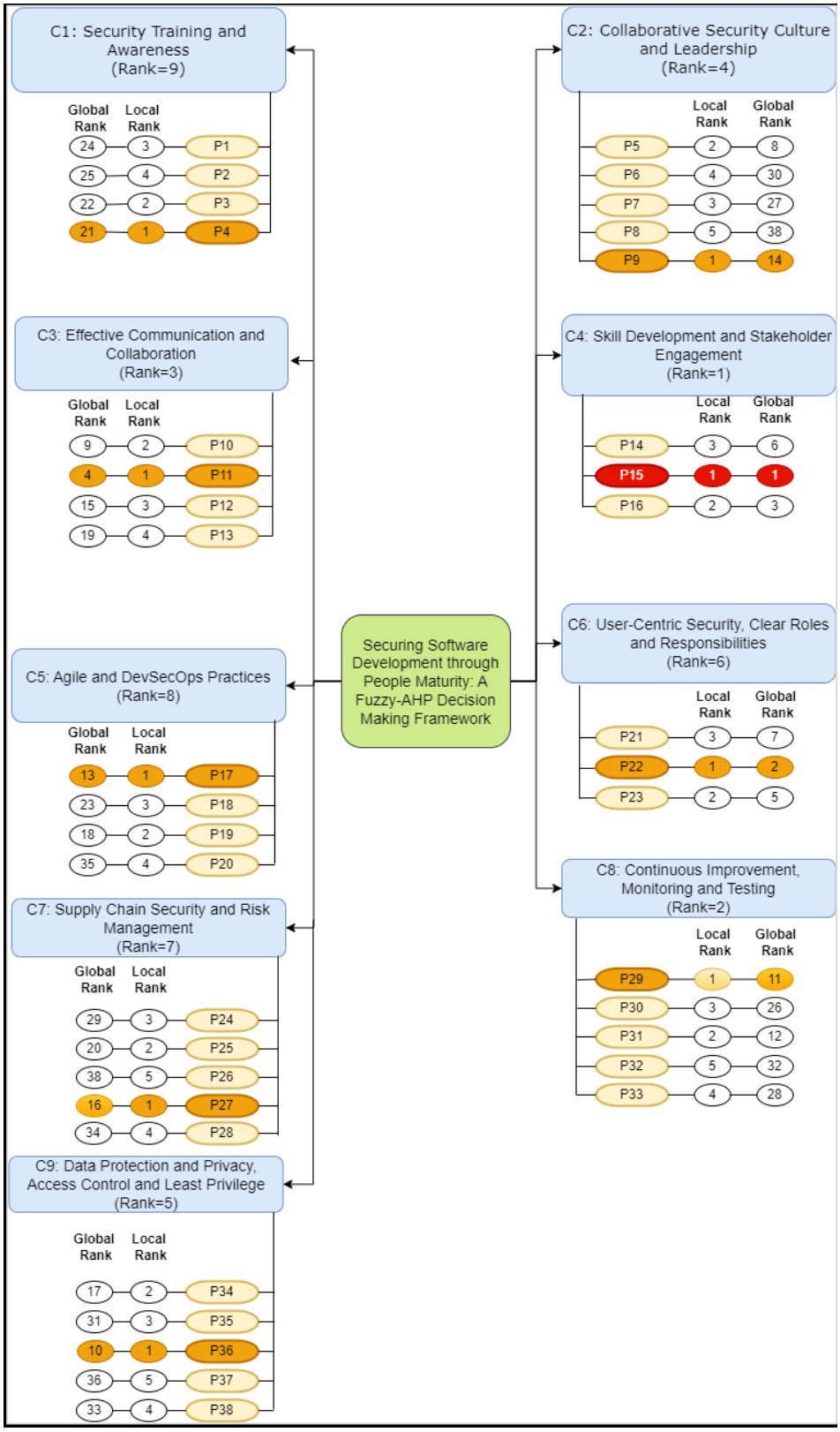

Figure 8. Securing Software Development through People Maturity: A Fuzzy-AHP Decision-Making Framework.

6. Implications of the Study

This research holds several implications for both academic professionals and practitioners within the software security domain:

 doi:10.20944/preprints202407.0669.v1

- **For Academics:** This study consolidates a comprehensive overview of HSFs and HSVs and their practices in SSD projects. Meticulously reviewing academic and published materials contributes to a valuable knowledge base. Researchers can leverage these findings to establish practical SSD projects, enhancing secure software development strategies. This study encourages SSD academia to incorporate the most prevalent HSFs and HSVs and their practices for SSD in their ongoing and future projects.
- **For Practitioners:** Industry experts gain insights into HSFs and HSVs and their practices in secure software development through an extensive literature review and empirical studies. Identifying 12 HSFs and 12 HSVs and 38 prioritized practices for SSD processes, this study aids professionals in implementing robust security measures. By addressing and mitigating these HSFs and HSVs, practitioners can refine existing techniques and devise novel approaches for the seamless adoption of HSFs and HSVs in SSD practices. The comprehensive overview offered by this study assists practitioners in recognizing critical security HSF and HSV practices pivotal for SSD projects.

The outcomes and recommendations derived from this framework significantly contribute to enhancing security in software development projects. Thus, the mentioned proposed framework is an effective instrument for development of new approaches and enhancing the SSD project quality by the assistance of both industry specialists and academic researchers.

The identified priorities give the HSF and HSV practitioners direction to encourage them to select the most appropriate practice depending on their practice categories and their general influence on the SSD context. This prioritization mechanism is thought to help SSD teams and researchers to deal with some of the most typical HSFs and HSVs experienced by programs and organizations to which SSD belongs.

Altogether, this is a comprehensive investigation of the HSFs and HSVs along with related practices in the context of the SSD system and contributes to the lack of literature in this area. The findings are useful for SDPs planning to implement HSFs and HSVs practices in the organizations that work on secure software development.

7. Summary and limitation of the study

This study aims to discern HSFs and HSVs and its practices for SSD projects and formulate a people maturity decision-making framework for secure software development, enhancing software development initiatives. Initially, 24 HSFs and HSVs were identified along with 38 practices to mitigate them through empirical studies involving 70 SSD experts. Additionally, 20 SSD expert panels were convened to analyze the interrelationship between HSF and HSV practices using the FAHP method. Based on collected data, FAHP prioritized these practices, highlighting the most critical category, “C4: Skill Development and Stakeholder Engagement,” which ranked highest at rank-1 and possessed the most significant weight of 0.12435, as detailed in Table 24. Similarly, the highest global weight, 0.051506, and global ranked (rank-1) HSF and HSV practice are “P15: Hands-On Practice and Stakeholder Communication,” as detailed in Table 24.

This study utilized an empirical questionnaire survey to explore people's maturity towards SSD by identifying HSFs and HSVs and their practices, ensuring content validity by aligning the survey with prior SLR findings. Construct validity was confirmed by survey respondents using various scales to assess attribute relevance. Internal validity leveraged SLR results to structure the questionnaire, while external validity incorporated a diverse group of international security experts who participated voluntarily, ensuring varied industry and project backgrounds. The small sample size in the fuzzy-AHP analysis could potentially limit findings due to the interpretive nature of this method.

 doi:10.20944/preprints202407.0669.v1

8. Conclusion and Future Direction

The study advocates integrating security measures for SSD projects, emphasizing the importance of addressing HSFs and HSVs and their practices. This empirical study fills a void by delving into HSFs and HSVs, their specific practices, and their interrelationships, creating a framework via the FAHP approach. The taxonomy developed offers valuable insights for academic researchers and practitioners, guiding prioritization and fostering innovation in SSD processes. The proposed framework aids SSD organizations in assessing and enhancing people's maturity for secure software development activities.

The future goal is to carry out more extensive empirical tests and real-life applications of the proposed framework. The rationale is to extend the usefulness of the presented approach for various software development settings and project sizes. Furthermore, the plan is to enhance people's decision-making processes by applying the Fuzzy-AHP method and integrating it with state-of-the-art machine learning algorithms. Thus, these accomplishments aim to offer a better and more responsive solution in terms of promoting maturity rationale for software development teams and boosting the overall level of security and quality within projects. Future research will further investigate the effects of the proposed framework on the aspects of teamwork, efficiency, and success of the projects to set its applicability as the norm in the field of secure software development.

This study also envisions designing a software product resilient to HSFs and HSVs and their practices and establishing testing protocols for such factors. A comparative analysis will be conducted, pitting our framework against other security models/frameworks in a hybrid approach.

Author Contributions: All the authors contributed equally in this article.
Funding: "This research received no external funding".
Data Availability Statement: All the data are presented in this article.
Acknowledgments: The researchers thank the Deanship of Scientific Research at King Saud University and the University of Malakand, Pakistan, for supporting this research.
Conflicts of Interest: Declare conflicts of interest or state "The authors declare no conflicts of interest."

References

1. Del-Real, C., E. De Busser, and B. van den Berg, *Shielding software systems: A comparison of security by design and privacy by design based on a systematic literature review.* Computer Law & Security Review, 2024. **52**: p. 105933.
2. Jørgensen, M. and E. Escott, *Relative estimates of software development effort: Are they more accurate or less time-consuming to produce than absolute estimates, and to what extent are they person-independent?* Information and Software Technology, 2022. **143**: p. 106782.
3. Khan, R.A., et al., *Evaluation of requirement engineering best practices for secure software development in GSD: An ISM analysis.* Journal of Software: Evolution and Process, 2024. **36**(5): p. e2594.
4. Barros, L., C. Tam, and J. Varajão, *Agile software development projects–Unveiling the human-related critical success factors.* Information and Software Technology, 2024. **170**: p. 107432.
5. Jadhav, A. and S.K. Shandilya, *Reliable machine learning models for estimating effective software development efforts: A comparative analysis.* Journal of Engineering Research, 2023. **11**(4): p. 362-376.
6. Nurwidyantoro, A., et al., *Human values in software development artefacts: A case study on issue discussions in three Android applications.* Information and Software Technology, 2022. **141**: p. 106731.
7. Adolph, S., P. Kruchten, and W. Hall, *Reconciling perspectives: A grounded theory of how people manage the process of software development.* Journal of Systems and Software, 2012. **85**(6): p. 1269-1286.
8. Anu, V., K.Z. Sultana, and B.K. Samanthula. *A Human Error Based Approach to Understanding Programmer-Induced Software Vulnerabilities*. in *2020 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW)*. 2020.
9. Fontana, R.M., et al., *Processes versus people: How should agile software development maturity be defined?* Journal of Systems and Software, 2014. **97**: p. 140-155.
10. Martins, J., F. Branco, and H. Mamede, *Combining low-code development with ChatGPT to novel no-code approaches: A focus-group study.* Intelligent Systems with Applications, 2023. **20**: p. 200289.
11. Pottebaum, J., et al. *Re-Envisioning Industrial Control Systems Security by Considering Human Factors as a Core Element of Defense-in-Depth*. in *2023 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)*. 2023.

12. Fujdiak, R., et al. *Managing the Secure Software Development*. in *2019 10th IFIP International Conference on New Technologies, Mobility and Security (NTMS)*. 2019.
13. Averin, A. and N. Zyulyarkina. *Characteristics of a Random Vector Obtained Using Human-Computer Interaction*. in *2020 Global Smart Industry Conference (GloSIC)*. 2020.
14. Zhanwei, H., et al. *Software security testing based on typical SSD:A case study*. in *2010 3rd International Conference on Advanced Computer Theory and Engineering(ICACTE)*. 2010.
15. Khan, R.A., et al., *Systematic Mapping Study on Security Approaches in Secure Software Engineering.* IEEE Access, 2021. **9**: p. 19139-19160.
16. van Solingen, R., et al., *From process improvement to people improvement: enabling learning in software development.* Information and Software Technology, 2000. **42**(14): p. 965-971.
17. Khan, R.A., et al., *Security Assurance Model of Software Development for Global Software Development Vendors.* IEEE Access, 2022. **10**: p. 58458-58487.
18. Khan, R.A., et al., *Systematic Literature Review on Security Risks and its Practices in Secure Software Development.* IEEE Access, 2022. **10**: p. 5456-5481.
19. Alzahrani, A. and R.A. Khan, *Secure software design evaluation and decision making model for ubiquitous computing: A two-stage ANN-Fuzzy AHP approach.* Computers in Human Behavior, 2024. **153**: p. 108109.
20. Khan, R.A., et al., *Security risks of global software development life cycle: Industry practitioner's perspective.* Journal of Software: Evolution and Process, 2024. **36**(3): p. e2521.
21. Khan, R.A. and S.U. Khan, *A preliminary structure of software security assurance model*, in *Proceedings of the 13th International Conference on Global Software Engineering*. 2018, Association for Computing Machinery: Gothenburg, Sweden. p. 137–140.
22. Khan, R.A., et al., *The State of the Art on Secure Software Engineering: A Systematic Mapping Study*, in *Proceedings of the 24th International Conference on Evaluation and Assessment in Software Engineering*. 2020, Association for Computing Machinery: Trondheim, Norway. p. 487–492.
23. Khan, R.A., S.U. Khan, and M. Ilyas, *Exploring Security Procedures in Secure Software Engineering: A Systematic Mapping Study*, in *Proceedings of the 26th International Conference on Evaluation and Assessment in Software Engineering*. 2022, Association for Computing Machinery: Gothenburg, Sweden. p. 433–439.
24. Alnaseef, F., et al., *Towards a successful secure software acquisition.* Information and Software Technology, 2023. **164**: p. 107315.
25. Hofmans, L., A. van der Stappen, and W. van den Bos, *Developmental structure of digital maturity.* Computers in Human Behavior, 2024. **157**: p. 108239.
26. Jukić, T., et al., *Organizational maturity for co-creation: Towards a multi-attribute decision support model for public organizations.* Government Information Quarterly, 2022. **39**(1): p. 101623.
27. Koolen, C., et al., *From insight to compliance: Appropriate technical and organisational security measures through the lens of cybersecurity maturity models.* Computer Law & Security Review, 2024. **52**: p. 105914.
28. Kadenic, M.D., K. Koumaditis, and L. Junker-Jensen, *Mastering scrum with a focus on team maturity and key components of scrum.* Information and Software Technology, 2023. **153**: p. 107079.
29. Glasauer, C., *The Prevent-Model: Human and Organizational Factors Fostering Engineering of Safe and Secure Robotic Systems.* Journal of Systems and Software, 2023. **195**: p. 111548.
30. Abad-Segura, E., et al., *Influential factors for a secure perception of accounting management with blockchain technology.* Journal of Open Innovation: Technology, Market, and Complexity, 2024. **10**(2): p. 100264.
31. De Win, B., et al., *On the secure software development process: CLASP, SDL and Touchpoints compared.* Information and Software Technology, 2009. **51**(7): p. 1152-1171.
32. Kitchenham, B., et al., *Systematic literature reviews in software engineering – A systematic literature review.* Information and Software Technology, 2009. **51**(1): p. 7-15.
33. Kitchenham, B. and S. Charters, *Guidelines for performing systematic literature reviews in software engineering.* 2007.
34. Dissanayake, N., et al., *Software security patch management - A systematic literature review of challenges, approaches, tools and practices.* Information and Software Technology, 2022. **144**: p. 106771.
35. Shukla, A., et al., *System security assurance: A systematic literature review.* Computer Science Review, 2022. **45**: p. 100496.
36. Kitchenham, B., *Procedures for performing systematic reviews.* Keele, UK, Keele University, 2004. **33**(2004): p. 1-26.
37. Lethbridge, T.C., S.E. Sim, and J. Singer, *Studying Software Engineers: Data Collection Techniques for Software Field Studies.* Empirical Software Engineering, 2005. **10**(3): p. 311-341.
38. Creswell, J.W., *Research design: qualitative, quantitative and mixed methods approaches, 3rd edition*. 2009: Sage, London.
39. Wagner, S., et al., *Status Quo in Requirements Engineering: A Theory and a Global Family of Surveys.* ACM Trans. Softw. Eng. Methodol., 2019. **28**(2): p. Article 9.
40. Humayun, M., et al., *Secure Global Software Development: A Practitioners’ Perspective.* Applied Sciences, 2023. **13**(4): p. 2465.

41. Ilyas, M., et al., *Software integration model: An assessment tool for global software development vendors.* Journal of Software: Evolution and Process. **n/a**(n/a): p. e2540.
42. Kitchenham, B. and S.L. Pfleeger, *Principles of survey research part 6: data analysis.* SIGSOFT Softw. Eng. Notes, 2003. **28**(2): p. 24–27.
43. Zadeh, L.A., *Fuzzy Sets, Fuzzy Logic, and Fuzzy Systems.* Advances in Fuzzy Systems — Applications and Theory, 1996. **6**: p. 1-840.
44. Marimon, F., M. Casadesus, and I. Heras-Saizarbitoria, *ISO 9000 and ISO 14000 standards: An international diffusion model.* International Journal of Operations & Production Management, 2006. **26**: p. 141-165.
45. Ayhan, M., *A Fuzzy AHP Approach for Supplier Selection Problem: A Case Study in a Gear Motor Company.* International Journal of Managing Value and Supply Chains, 2013. **4**.
46. Chamodrakas, I., D. Batis, and D. Martakos, *Supplier selection in electronic marketplaces using satisficing and fuzzy AHP.* Expert Systems with Applications, 2010. **37**(1): p. 490-498.
47. Chang, D.-Y., *Applications of the extent analysis method on fuzzy AHP.* European Journal of Operational Research, 1996. **95**(3): p. 649-655.
48. Stelzer, D. and W. Mellis, *Success factors of organizational change in software process improvement.* Software Process: Improvement and Practice, 1998. **4**(4): p. 227-250.
49. Corbin, J.M. and A. Strauss, *Grounded theory research: Procedures, canons, and evaluative criteria.* Qualitative sociology, 1990. **13**(1): p. 3-21.
50. Khan, A.A., et al., *Agile trends in Chinese global software development industry: Fuzzy AHP based conceptual mapping.* Applied Soft Computing, 2021. **102**: p. 107090.
51. Khan, R.A., et al., *Security risks of global software development life cycle: Industry practitioner's perspective.* Journal of Software: Evolution and Process, 2023. **n/a**(n/a): p. e2521.
52. Khan, R.A., et al., *An Evaluation Framework for Communication and Coordination Processes in Offshore Software Development Outsourcing Relationship: Using Fuzzy Methods.* IEEE Access, 2019. **7**: p. 112879-112906.
53. Yaghoobi, T., *Prioritizing key success factors of software projects using fuzzy AHP.* Journal of Software: Evolution and Process, 2017. **30**: p. e1891.
54. A. K. Barbara, B. D. Budgen, and O.P. Brereton, *Using mapping studies as the basis for further research A participant-observer case study.* Information Software Technology, 2011. **53**(6): p. 638-651.
55. Admass, W.S., Y.Y. Munaye, and A.A. Diro, *Cyber security: State of the art, challenges and future directions.* Cyber Security and Applications, 2024. **2**: p. 100031.
56. Li, B., et al., *Cognitively reconfigurable mimic-based heterogeneous password recovery system.* Computers & Security, 2022. **116**: p. 102667.