Incorporating Cybersecurity Requirements in Agile Development Processes

Gabriella Castro Barbosa Costa Dalpra[1,2][0000−0003−2236−3825],
Victor Borges Loures de Paula[1][0009−0002−8710−0815],
Marcelo Augusto Silva Belisário[1][0009−0003−8974−0258],
Maria Júlia Marques Schettini[1][0009−0003−3261−2490],
José Eduardo Fernandes[2][0000−0001−9638−7593], and
Tiago Pedrosa[2,3][0000−0003−4873−2705]

[1] Federal Center for Technological Education of Minas Gerais
Computer Department, Leopoldina, 36700-001, Brazil
gabriella@cefetmg.br, {vborgescontato,marcelobeli9,
mariajuliaschettini97}@gmail.com
[2] Research Centre in Digitalization and Intelligent Robotics (CeDRI),
Instituto Politécnico de Bragança, 5300-253 Bragança, Portugal
{jef,pedrosa}@ipb.pt
[3] Laboratório Associado para a Sustentabilidade e Tecnologia em Regiões de Montanha (SusTEC), Instituto Politécnico de Bragança, 5300-253 Bragança, Portugal
pedrosa@ipb.pt

Abstract. The complexity and interconnectivity of modern software systems make cybersecurity a key concern. Each new device connected to a network increases security risks, and the rise of cyber threats and attack sophistication requires strong security measures from the start of development. Agile development methodologies are widely adopted for their ability to improve efficiency and flexibility in software development. However, these methodologies often lack clear guidance on how to incorporate security practices effectively. This gap can result in vulnerabilities being exploited by attackers, compromising the security of sensitive systems and data. Based on this fact, balancing security with agility is essential in today's fast-paced digital landscape. Considering the application of cybersecurity requirements in agile development processes, since the beginning of the software development lifecycle, we propose the ASF Framework, developed using the Design Science Research approach. The framework evaluation focuses on its application to the OWASP ASVS cybersecurity standard and Scrum, using user stories with Acceptance Criteria and the Definition of Done to ensure clear and measurable development goals. This assessment was conducted in the context of a web platform that connects consumers and service providers, simplifying the process of offering and hiring services. The evaluation demonstrates the applicability of the ASF Framework in a real-world scenario, and the results indicate that ASF effectively supported the identification of security requirements within an agile development context.

Keywords: Cybersecurity · Agile Methodologies · Security Requirements.

Proceedings of the 28th Workshop on Requirements Engineering (WER2025),
August 20-22, 2025, Rio de Janeiro-RJ, Brazil.
https://doi.org/10.29327/1588952.28-13

1 Introduction

While security requirements encompass protections against a broad range of threats, including accidental and intentional ones, cybersecurity requirements are a subset of security requirements, focusing specifically on defending systems against malicious digital attacks [1]. Thus, the increasing complexity and interconnectivity of modern software systems have made cybersecurity a central concern for developers and businesses worldwide. Every new device connected to a network raises security risks [2], and the growing proliferation of cyber threats, along with the sophistication of attacks, demands the implementation of robust security measures from the earliest stages of software development. Besides that, many organizations, companies, and governments lack experienced personnel in the cybersecurity domain, having difficulties in adopting a standard approach or a cybersecurity framework [3].

Agile development methodologies such as Scrum[4] and XP [5] are widely adopted for their ability to improve efficiency and flexibility in software development. However, integrating cybersecurity requirements with agile development methodologies remains a significant challenge: these methodologies often lack clear guidance on how to incorporate security practices effectively, and this gap can result in vulnerabilities being exploited by attackers, compromising the security of the systems. Valdés-Rodrígues et al. [6] underscore the necessity of addressing security in the agile environment, as it remains a significant challenge in software development. Then, this integration is crucial to ensure that systems not only meet market demands for functionality and speed but are also protected against vulnerabilities that could compromise data and operations. Balancing security with agility is essential in today's fast-paced digital landscape.

The main goal of this paper is to present a framework that **incorporates cybersecurity requirements specification in agile development processes**, addressing the critical need to integrate security from the beginning of the software development lifecycle. It aims to provide concrete and applicable guidelines without compromising the agility required for the development process.

The framework was developed using the Design Science Research (DSR) approach [16]. When using DSR, the researcher explores artifacts and natural settings by formulating hypotheses (design), conducting experiments (instantiating artifacts), and comparing results with expectations (evaluation). Initially, we conducted a literature review to study research topics and a gap considering the application of cybersecurity standards in agile software development (detailed in Section 3). Then, we defined the research problem (*How to implement cybersecurity in the context of agile software development, since the beginning of the software development lifecycle?*) and proposed our framework. The framework evaluation considers its application to the OWASP ASVS [7] cybersecurity standard, the standard that appeared most in our literature review, and also the most widely used agile development methodology (Scrum), using User Stories with Acceptance Criteria (AC) [8] and considering the concept of Definition of Done (DoD) [9]. Despite that, it should be considered that the proposed framework is capable of meeting other cybersecurity standards and also

other agile methodologies that work with the specification of work items and/or backlog items.

The remainder of this paper is structured as follows. Section 2 details the concepts used in the definition of the ASF Framework, followed by Related Work. Section 4 describes the ASF. The framework in action is presented in Section 5, and, finally, Section 6 provides the conclusion, followed by the references.

2 Background

2.1 Agile Software Development and Scrum Concepts

The "Manifesto for Agile Software Development" [10] outlines four core values of Agile Software Development: prioritizing (i) individuals and interactions over processes and tools, (ii) working software over comprehensive documentation, (iii) customer collaboration over contract negotiation, and (iv) responding to change over following a rigid plan. At its heart, this manifesto emphasizes lightweight yet sufficient project management rules, with a strong focus on human-centric and communication-driven practices [11]. Agile stands as the most widely adopted software development methodology [12], with numerous techniques and frameworks developed to support its principles (e.g., Agile Unified Process (AUP), eXtreme Programming (XP), Lean Agile, Microsoft SDL for Agile, Scaled Agile Framework (SAFe)). Among these, Scrum [4] is the most popular team-level methodology: 63% of Agile users are team Scrum [13].

The Scrum framework [14] has as roles a Scrum Team consisting of a **Product Owner**, a **Scrum Master**, and **Developers**, each of which has specific responsibilities. The Scrum Team participates in five events (The **Sprint**, **Sprint Planning**, **Daily Scrum**, **Sprint Review**, and **Sprint Retrospective**) and produces three artifacts (**Product Backlog**, **Sprint Backlog**, **Increment**). Taking into account that the case study for validating the framework presented in this work involves its application to Scrum, with direct impacts on the specification of the **Product Backlog** or during the Product Backlog item refinement, it is presented in detail in the following.

The **Product Backlog** is a dynamic, prioritized list of everything needed to enhance the product. Product Backlog items that can be completed within a Sprint are considered ready for selection during Sprint Planning.

Although there are multiple ways to write and format Product Backlog items, one of the most popular and effective approaches is using the user story format [8]. Despite its simple structure: As a *[who]*, I want *[what]*, so that *[why]*, it is possible to incorporate to the user story additional information in the form of AC. AC can be written as a series of bullet points as a list, or in whatever format is most useful for the Scrum team. Many teams choose to write the AC as a list of requirements that are associated with the story. In this vein, our applied case study uses this approach to incorporate most of the cybersecurity requirements into the product backlog items/user stories.

Finally, another important concept used to apply the approach in the context of Scrum is the DoD [9]. It is a formal description of the state of the Increment

when it meets the quality measures required for the product. DoD differs from user stories considering that DoD applies to the Increment, not directly to the product backlog items (or user stories in our case).

2.2 Cybersecurity Standards and Frameworks

Cybersecurity is the "safeguarding of people, society, organizations, and nations from cyber risks", while "safeguarding means to keep cyber risk at a tolerable level" [15].

Regarding the standards or frameworks related to cybersecurity, Syafrizal et al. [3] present a literature review with various types of cybersecurity standards and frameworks and their respective elements. The main difference between them is that a standard specifies what must be done to comply with the standard - by explaining and providing methods one by one to complete the process; while a framework provides general guidelines that organizations can adopt, covering multiple components or domains without prescribing exact steps.

Considering the cybersecurity frameworks or standards in the context of software development, according to a previous literature review, the following can be cited:

- **Security Engineering Framework (SEF)**[19]: it is a set of software-focused engineering practices for managing security and resilience risks throughout a system's lifecycle. It offers a roadmap for integrating security and resilience into software-driven systems and maintaining these capabilities during operations and sustainment. SEF structures its practices into domains and goals, providing detailed guidance for each.
- **OWASP SAMM** (Software Assurance Maturity Model) [17]: is a framework designed to help organizations create and implement a software security strategy tailored to their specific risks. In summary, it enables the organizations to: (i) Assess current software security practices; (ii) Develop a structured security assurance program in iterative steps; (iii) Show measurable improvements in security assurance; (iv) Define and track security activities across the organization.
- **OWASP ASVS** (Application Security Verification Standard) [7]: it provides a list of requirements for secure development and a basis for testing web application technical security controls.
- **NIST Cybersecurity Framework (CSF)** [18]: it is a framework that guides industries, government agencies, and organizations to manage cybersecurity risks. It defines key cybersecurity outcomes that any organization—regardless of size, sector, or maturity—can use to assess, prioritize, and communicate its cybersecurity efforts.
- **Microsoft Security Development Lifecycle (SDL)**[20]: it is an approach to integrate security into DevOps processes (sometimes called a DevSecOps approach), but it is cited that "the practices described in the SDL approach can be applied to all types of software development and all platforms from classic waterfall through to modern DevOps approaches". The 10 security

practices proposed by this approach to be integrated into the development processes are: (i) Establish security standards, metrics, and governance; (ii) Require use of proven security features, languages, and frameworks; (iii) Perform security design review and threat modeling; (iv) Define and use cryptography standards; (v) Secure the software supply chain; (vi) Secure the engineering environment; (vii) Perform security testing; (viii) Ensure operational platform security; (ix) Implement security monitoring and response; (x) Provide security training.

- **ISO/IEC 27005:2022** - Information security, cybersecurity, and privacy protection — Guidance on managing information security risks [21]: provides guidance to assist organizations to fulfil the requirements of ISO/IEC 27001 concerning actions to address information security risk and to perform information security risk management activities, specifically information security risk assessment and treatment. It is cited that this standard "applies to all organizations, regardless of type, size or sector".
- **Oracle Software Security Assurance (OSSA)** [22]: is an Oracle methodology to embedding security at every stage of product development - design, build, testing, and maintenance.

Other cybersecurity standards, frameworks, or regulations that can be cited are: ISO/IEC 25010:2011, Team Software Process Secure (TSP-Secure), Software Assurance Maturity Model (OpenSAMM) from OWASP, Building Security In Maturity Model Framework (BSIMM), and IEC 62443-4-1.

Considering that the OWASP Foundation has a specific project that provides developers with a list of requirements for secure development, the OWASP Application Security Verification Standard (ASVS) was selected to apply the proposed approach. As presented before, OWASP ASVS provides "a list of application security requirements or tests that can be used by architects, developers, testers, security professionals, tool vendors, and consumers to define, build, test, and verify secure applications".

3 Related Work

The effective integration of security requirements and agile methodologies requires not only the recognition of their importance but also the implementation of specific procedures / practices to ensure the protection and integrity of the developed software [6] and this framework fits in this context. Related works will be described considering the use of Scrum [23,26,28] or the OWASP [25,27,24].

Azham et al. [23] use Scrum as the basis and propose the integration of security principles in development phases, and implement a new backlog, called *Security Backlog*. It produces a simple but comprehensive document to address the security risks. A new role is proposed, named *Security Master*, which is responsible for security risks, feature identification, testing, and documentation during the development process. Different from Azham et al. [23], our proposed framework does not create another backlog specifically to deal with security requirements and doesn't establish another role to deal with it.

Yee [23] describes an approach called VMASIR - The Visual Model for Attack Surface Identification and Reduction, which is a visual modeling approach (inspired by the Data Flow Diagram) to identify and reduce the attack surface in software systems that contain sensitive data. Despite it is suitable to be used with Scrum, the approach deals directly with identifying and reducing the attack surface and not with prescribing user stories with security requirements.

Asprion et al. [28] present the M4ACSM - Model for Agile Cybersecurity Management - a model for managing cybersecurity with an agile approach. M4ACSM combined the NIST CSF [18] with Agile/Scrum by adding the NIST CSF core functions, the NIST CSF´s implementation steps with agile procedures and artifacts. However, it does not provide direct and specific instructions on how to deal with cybersecurity requirements.

Considering the approaches that cite the OWASP standard, Villamizar et al. [25] present an approach to address security in the early stages of the software development lifecycle, providing a focused reading technique that can be used to support the manual inspection of agile specifications. This proposal also deals with user stories, considers an older version of OWASP ASVS, identifies high-level security requirements from the OWASP to be verified, and generates a reading technique to support reviews in detecting defects. The approach is detailed and also validated with a controlled experiment in academic settings. The main differences between this work and our proposal are based on the consideration of Scrum as our framework and also the focus of the approach - while Villamizar et al. focused on proposing and evaluating an approach for reviewing security-related aspects in agile requirements specifications of web applications, our focus is on security requirements specification.

Núñez et al. [27] proposes the Viewnext-UEx model - a preventive and flexible Secure Software Development Model, organized into four areas of development (Policies, Secure Development Methodology, Supervision, and Observatory) and fourteen security activities: (1) Strategy and orientation, (2) Training, (3) Definition of risk, (4) Requirements validation, (5) Threat modeling, (6) Design review, (7) Development revision, (8) Security testing, (9) Output validation, (10) Evaluations and metrics, (11) State of project, (12) Response plan and incidents, (13) Security observatory, (14) Vulnerabilities repository. The Secure Development Methodology area encompasses activities 4 to 9 and during activity (6) Design review, the design and architecture of the software are evaluated based on OWASP Application Security Verification Standard to detect security-related problems. This activity is performed before starting the development, avoiding potential costs of solving security problems. It differs from our approach, which considers using OWASP during the software development (using Scrum) and not only before starting the development.

Finally, Othmane et al. [24] present a method for security assurance of software increments that integrates security engineering activities into the agile software development process. Considering the inception phase, they suggest that the team members compose a set of security goals that address the threats and the security goals take the form of claims for the assurance cases and are added as

security user stories to the project backlog. Different from them, we do not create specific stories to deal with the threats; we incorporate the security requirements into the user stories as AC.

4 Framework Definition

The ASF Framework was developed considering the Design Science Research (DSR) approach [16]. DSR is based on the principle that knowledge and understanding of a design problem and its solution are achieved through the creation and application of an artifact. This process involves developing a purposeful artifact, tailored to a specific problem domain, ensuring it has utility, quality, and efficacy, demonstrated via well-executed evaluation methods. The artifact must also provide clear contributions in the areas of the design artifact and require the application of rigorous methods in its construction and evaluation. The methodology is inherently iterative; the search for an effective artifact requires utilizing available means to reach desired ends while satisfying laws in the problem environment, and, finally, the outcomes of the research must be communicated clearly and effectively to facilitate implementation and inspire further exploration.

Considering the research problem (*How to incorporate cybersecurity requirements specification in the context of agile software development, since the beginning of the software development lifecycle?*), ASF was defined based on the execution of six tasks, that should be started before the first iteration of the software product to be developed. It should be noted that the framework may even be implemented for software products that are already under development or with previous versions running/in production, however, it will not be possible to guarantee that the increments that already exist before the framework usage will be in accordance with the chosen cybersecurity standard. These framework-required tasks are presented in Figure 1 and detailed in the following:

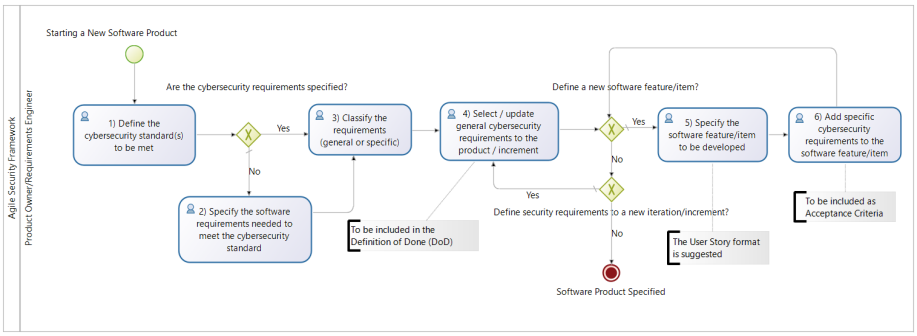

Fig. 1. ASF Framework Tasks

1. **Define the cybersecurity standard(s) to be met**: Although the proposed framework was initially developed and evaluated to be used in conjunction with the security requirements specified by the OWASP ASVS standard (as presented in Section 4), it is understood that the set of activities presented by the ASF Framework can be applied to any standards that address cybersecurity issues, as long as the specifications of these standards are in the form of software requirements or can be transformed into software requirements. The adoption of OWASP ASVS Level 1 requirements was chosen because this level includes fundamental security controls, these being the minimum requirements that all applications must seek, which facilitates their verification and application in agile projects, as they can be verified automatically, without access to the source code. An example of an OWASP ASVS cybersecurity requirement is: "*Verify that all keys and passwords are replaceable and are part of a well-defined process to re-encrypt sensitive data*".
2. **Specify the software requirements needed to meet the cybersecurity standard**: This is an optional task, and it needs to be executed only if the standard does not provide the cybersecurity requirements already specified.
3. **Classify the requirements (general or specific)**: Considering the cybersecurity standards to be met, it is necessary to classify them as to whether they are **specific** to a software item to be implemented (currently the framework has been validated considering user stories) or whether this requirement should be considered for the increment as a whole, in **general**. For example: the requirement '*Verify that all components are up to date, preferably using a dependency checker during build or compile time*' should be considered and validated for all user stories, and not just for a specific story. The requirement '*Verify that user set passwords are at least 12 characters in length (after multiple spaces are combined)*', on the other hand, should only be considered for user stories that involve developing some function that deals with user passwords.
4. **Select/update general cybersecurity requirements to the product/increment**: During this task, generic security requirements (as established in task **3. Classify the requirements (general or specific)**) must be selected and applied to the increment as a whole and not just to a specific user story, through the Increment DoD list. In the case of defining a new increment, in a new iteration to be carried out, this list can be readjusted according to the increment to be generated.
5. **Specify the software feature/item to be developed**: Software items to be developed during the iteration (Sprint) should be specified during this task. Our suggestion is to do this specification using the user story format: *As a **[who]**, I want **[what]**, so that **[why]**.* Initially, this specification of the features to be developed must follow the same rhythms as the agile methodology in question. The difference lies in the next activity to be performed.
6. **Add specific cybersecurity requirements to the software feature/item**: During this task, specific security requirements (as established in task **3. Classify the requirements (general or specific)**) must be selected and applied to each specific user story as some AC. If there are many security

requirements to be checked for each user story, it is suggested to group them according to their applicable context to simplify this definition and speed up the process.

All the framework activities are suggested to be done by the requirements engineering or by the Product Owner (if the agile methodology in use is Scrum).

5 Framework in Action

To showcase the applicability of ASF, we applied it to a web system designed to serve as a digital platform connecting consumers with service providers. This platform streamlines the process of hiring and offering services, ensuring efficiency and organization. The platform enables users to register as either consumers or service providers, allowing consumers to search for professionals, request services, and provide feedback, while service providers can manage their availability, accept or decline service requests, and receive ratings from customers. Additionally, the system includes features such as chat communication, media sharing, search filters, and a structured evaluation process to ensure a transparent and reliable service experience for all users. Considering the web system context, the tasks from ASF were executed as follows.

1. **Define the cybersecurity standard(s) to be met**: The framework evaluation focuses on its application to the OWASP ASVS cybersecurity standard — the most frequently cited standard in our literature review, considering its' first level;
2. **Specify the software requirements needed to meet the cybersecurity standard**: Considering that OWASP ASVS already provides a list of 278 application security requirements (128 related to the first level), up to the publication date of this paper, only the first level is applicable;
3. **Classify the requirements (general or specific)**: To do this task, a spreadsheet[4] was created considering all the 128 OWASP ASVS - level 1 requirements. This classification was carried out and verified jointly by a cybersecurity expert and a software engineering expert with more than ten years of experience in their areas.
4. **Select general cybersecurity requirements to the product/increment**: Considering the example used in this Section, all the 45 general requirements (as specified in the spreadsheet cited in the last task) should be applied to the DoD list of the web application. Although the number of requirements to be verified with each new increment of the product to be generated, a great advantage of these general requirements (which refer to level 1 of the OWASP ASVS) is that they can be tested through penetration tests, which can also be automated by organizations.
5. **Specify the software feature/item to be developed**:

[4] External spreadsheet: https://bit.ly/4bZf4aT

The first increment of the proposed system was specified by the following user stories: **(1)**: As a user, I want to become a service provider by registering on the platform, adding my available schedule, city of operation, professional profile description, and specialties, So I can sell my services. **(2)**: As a user, I want to register on the platform by entering my name, CPF, date of birth, email, phone number, password, profile picture link, and address (ZIP code, number, and complement), So I can hire service providers. **(3)**: As a user, I want to log in by entering my email and password, To access my personal account. **(4)**: As a user, I want to chat and attach documents such as photos, videos, and audio files, To showcase my work or explain the issue that needs to be resolved. **(5)**: As a consumer, I want to rate the service by giving 1 to 5 stars, leaving a comment, and adding photos, videos, and audio files, To provide feedback on the service. **(6)**: As a consumer, I want to search for professionals by typing the desired category in a search field with auto-complete, displaying results in alphabetical order regardless of relevance, To find the best service provider based on availability. **(7)**: As a consumer, I want to request a service by filling out a form containing the following details: service order number, service description, price, service location, customer information (name/legal entity, CPF/CNPJ, address, phone number, etc.), issue date, and payment method, To create a service request, allowing the provider to accept the job and solve my problems. **(8)**: As a service provider, I want to accept or decline a service request, which will expire after 24 hours, To have the option to choose jobs based on my availability. **(9)**: As a service provider, I want to rate customers by giving 1 to 5 stars, leaving a comment, and adding photos, videos, and audio files, To help other providers understand how the customer interacts with service providers and whether they follow the platform's guidelines. **(10)**: As a consumer, I want to apply search filters (such as availability and location) to refine my search, To make it easier to find the most suitable service efficiently and clearly.

6. **Add specific cybersecurity requirements to the software feature/item**: For each of the ten user stories listed above, all 83 specific cybersecurity requirements outlined in the external spreadsheet must be analyzed. However, this process can be highly time-consuming for the Product Owner (PO) or requirements engineer. To streamline this task, the requirements were categorized into groups (questions) and subgroups (subquestions), making them easier to manage. To further simplify this process, the ASF Tool was developed[5] to assist in defining cybersecurity requirements based on this classification. As shown in Figure 2, the responsible user inputs the user story into the system and is immediately guided to a questionnaire (as shown in Figure 3). This questionnaire dynamically filters the 83 requirements based on the responses to its questions and sub-questions, making the requirement selection process more efficient and precise. At the end of the process, the tool generates a list of selected security requirements for each user story to be directly supported as AC, and can be integrated

[5] `https://asf.linceonline.com.br/asf`

into a backlog management tool[6]. As an example, considering the user story 8: *As a service provider, I want to accept or decline a service request, which will expire after 24 hours, To have the option to choose jobs based on my availability.*", the following cybersecurity requirements were specified: **V3.2.2** - Verify that session tokens possess at least 64 bits of entropy; **V3.2.3** - Verify the application only stores session tokens in the browser using secure methods such as appropriately secured cookies (see section 3.4) or HTML 5 session storage; **V3.3.1** - Verify that logout and expiration invalidate the session token, such that the back button or a downstream relying party does not resume an authenticated session, including across relying parties; **V3.3.2** - If authenticators permit users to remain logged in, verify that re-authentication occurs periodically both when actively used or after an idle period; **V3.7.1** - Verify the application ensures a full, valid login session or requires re-authentication or secondary verification before allowing any sensitive transactions or account modifications; **V5.1.3** - Verify that all input (HTML form fields, REST requests, URL parameters, HTTP headers, cookies, batch files, RSS feeds, etc) is validated using positive validation (allow lists); **V8.3.4** - Verify that all sensitive data created and processed by the application has been identified, and ensure that a policy is in place on how to deal with sensitive data.

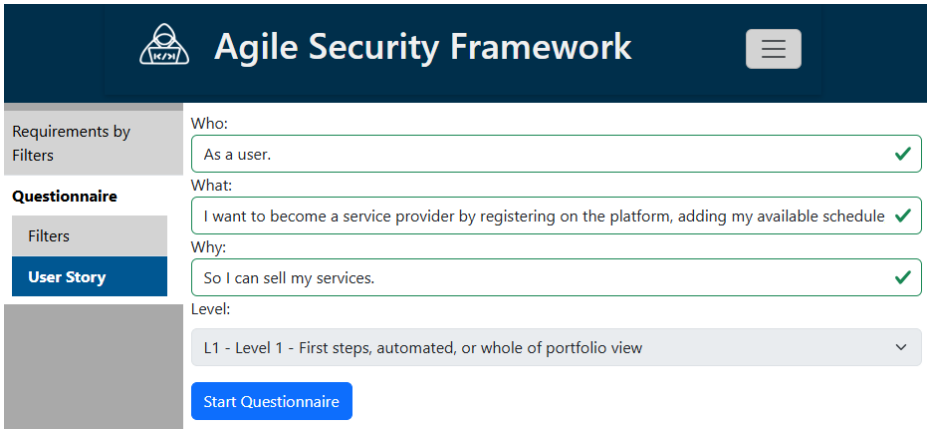

Fig. 2. ASF Tool Support - Story Definition

[6] The cybersecurity requirements defined for the 10 user stories (which should be attached as AC) are presented in the third tab of the external spreadsheet, aiming to demonstrate the applicability of the proposed framework.

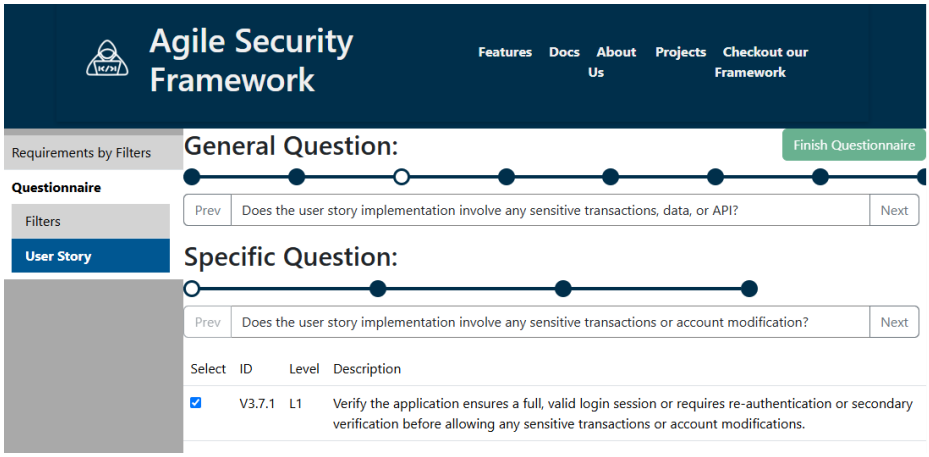

Fig. 3. ASF Tool Support - Story Cybersecurity Requirements Definition

6 Conclusion

The paper presents the ASF Framework that aims to integrate cybersecurity requirements into agile software development from the beginning of the development life cycle. The framework was evaluated considering its application to the OWASP ASVS standard and the Scrum methodology, using user stories with AC and the DoD. ASF has advantages such as (i) incorporating security requirements directly into user stories, avoiding the need for a separate backlog for security; (ii) using an automated questionnaire to facilitate the selection and application of relevant security requirements, reducing the workload of the Product Owner (PO) or requirements engineer; (iii) being adaptable to other security standards and agile methodologies that use backlog items. The application of the framework in a web system that connects consumers to service providers demonstrated its feasibility and benefits in incorporating security requirements without compromising development agility.

As threats to validity, the following can be cited: Considering Internal Validity: (i) The study focuses on the implementation of a single use case, and may not capture specific challenges of other types of systems; (ii) The categorization of security requirements was validated by only two experts, which may introduce subjective bias; (iii) The automated tool that assists in defining the requirements was not extensively tested in different development scenarios. As External Validity: (i) The framework was evaluated in a specific context (a web system for connecting consumers and providers), which may limit its applicability to other software domains; (ii) Compatibility with other security standards and agile methodologies was mentioned, but not tested in practice.

The application of the ASF Framework was presented, however, a more detailed evaluation, involving a greater number of experts and other systems,

is planned. Also, as future work, we could expand the ASF by testing it on different types of systems, as well as validating it with other security standards. Improvements to the automated tool include questionnaire optimization, integration with agile platforms, and usability assessment. Empirical and comparative studies could measure its impact on software productivity and security. In addition, automating security verification by integrating the ASF with analysis tools and DevSecOps pipelines could strengthen its effectiveness.

Acknowledgments. *We gratefully acknowledge the support provided by CEFET-MG, CNPq, and the LINCE laboratory. This work was also parcial supported by national funds: UID/05757 - Research Centre in Digitalization and Intelligent Robotics (CeDRI); and SusTEC, LA/P/0007/2020 (DOI: 10.54499/LA/P/0007/2020).*

References

1. National Institute of Standards and Technology (NIST). "Systems Security Engineering: Considerations for a Multidisciplinary Approach in the Engineering of Trustworthy Secure Systems," in NIST Special Publication 800-160, vol. 1. Gaithersburg, MD, 2016. https://doi.org/10.6028/NIST.SP.800-160v1.
2. Ahmad, N., LaPlante, P. A., DeFranco, J. F., and Kassab, M., *A cybersecurity educated community*, in IEEE Transactions on Emerging Topics in Computing, 10th ed. New York, NY, 2022. `https://doi.org/10.1109/TETC.2021.3093444`.
3. Syafrizal, M., Selamat, S. R., Zakaria, N. A. "Analysis of cybersecurity standard and framework components," in International Journal of Communication Networks and Information Systems. 2022. https://doi.org/10.17762/ijcnis.v12i3.4817.
4. Schwaber, K. *Agile Project Management with Scrum*. Microsoft Press, 2004.
5. Beck, K. *Extreme Programming Explained: Embrace Change*. Addison-Wesley, 1999.
6. Valdés-Rodríguez, Y., Hochstetter-Diez, J., Diéguez-Rebolledo, M., Bustamante-Mora, A., Cadena-Martínez, R. "Analysis of strategies for the integration of security practices in agile software development: A sustainable SME approach," in IEEE Access, New York, 2024. https://doi.org/10.1109/ACCESS.2024.3372385.
7. OWASP Foundation. *OWASP Application Security Verification Standard*. Available at: `https://owasp.org/www-project-application-security-verification-standard/`. Accessed on: jan. 2025.
8. Scrum.org. *User Story Format*. [Online]. Available at: `https://www.scrum.org/resources/blog/user-story-format`. Accessed on: jan. 2025.
9. Scrum.org. *Definition of Done (DoD): Explanation and Example*. [Online]. Available at: `https://www.scrum.org/resources/blog/definition-done-dod-explanation-and-example`. Accessed on: jan. 2025.
10. K. Beck, M. Beedle, A. Bennekum, et al., "Manifesto for Agile Software Development,". Available at: https://agilemanifesto.org/. Accessed on: jan. 2025.
11. A. Cockburn, "Agile Software Development," in Addison-Wesley, Boston, MA, 2002.
12. Chakravarty, K. and Singh, J. "A dissection of agile software development in changing scenario and the sustainable path ahead," in International Journal of System Assurance Engineering and Management, xth ed. 2024. `https://doi.org/10.1007/s13198-024-02283-1`.

13. PMwares. *17th Annual State of Agile Report*, 2023. Available at: https://digital.ai/resource-center/analyst-reports/state-of-agile-report/. Accessed on: jan. 2025.
14. Schwaber, K., and Jeff S. *The Scrum Guide: The Definitive Guide to Scrum: The Rules of the Game*. 2020. Available at: https://scrumguides.org/scrum-guide.html. Accessed: jan. 2025.
15. ISO/IEC 27032:2023; Information Technology and Security Techniques and Guidelines for Cybersecurity. Geneva, Switzerland: International Organization for Standardization; 2023.
16. Hevner, A., Salvatore, P., Sudha, R. Design science in information systems research, in: MIS Quarterly, Vol. 28, 2004, pp. 75–105.
17. OWASP Foundation. *OWASP SAMM*. [Online]. Available at: https://owasp.org/www-project-samm/. Accessed on: jan. 2025.
18. NIST. *The NIST Cybersecurity Framework (CSF) 2.0*.[Online]. Available at: https://doi.org/10.6028/NIST.CSWP.29. Accessed on: fev. 2025.
19. Alberts, C., Wallen, C., Woody, C., Bandor, M., Merendino, T. Security Engineering Framework (SEF): Managing Security and Resilience Risks Across the Systems Lifecycle. Carnegie Mellon University. 2024. https://doi.org/10.1184/R1/25029359
20. Microsoft SDL. *Microsoft Security Development Lifecycle (SDL)*.[Online]. Available at: https://www.microsoft.com/en-us/securityengineering/sdl?oneroute=true. Accessed on: fev. 2025.
21. ISO/IEC 27005:2022; Information security, cybersecurity and privacy protection — Guidance on managing information security risks. Available at: https://www.iso.org/obp/ui/en/#iso:std:iso-iec:27005:ed-4:v1:en. Accessed on: fev. 2025.
22. Oracle Software Security Assurance. Available at: https://www.oracle.com/corporate/security-practices/assurance/. Accessed on: fev. 2025.
23. Azham Z., Ghani, I., Ithnin, N. "Security backlog in Scrum security practices," in Proceedings of the 2011 Malaysian Conference in Software Engineering, xth ed. Kuala Lumpur, Malaysia, 2011:414-417. https://doi.org/10.1109/MySEC.2011.6140708.
24. Othmane, L. B., Angin, P., Weffers, H., Bhargava, B. "Extending the agile development process to develop acceptably secure software," in IEEE Transactions on Dependable and Secure Computing, xth ed. New York, NY, 2014;11(6):497-509. https://doi.org/10.1109/TDSC.2014.2298011.
25. Villamizar, H., Anderlin Neto, A., Kalinowski, M., Garcia, A., Mendez, D. "An approach for reviewing security-related aspects in agile requirements specifications of web applications," in Proceedings of the 27th IEEE International Requirements Engineering Conference (RE), xth ed. Jeju Island, South Korea, 2019:86-97. https://doi.org/10.1109/RE.2019.00020.
26. Yee, G. O. M. "Attack surface identification and reduction model applied in Scrum," in Proceedings of the 2019 International Conference on Cyber Security and Protection of Digital Services (Cyber Security), xth ed. Oxford, U.K., 2019:1-8. https://doi.org/10.1109/CyberSecPODS.2019.8884956.
27. Nunez, J. C. S., Lindo, A. C., Rodriguez, P. G. "A preventive secure software development model for a software factory: A case study," in IEEE Access, xth ed. 2020;8:77653-77665. https://doi.org/10.1109/ACCESS.2020.2989113.
28. Asprion, P. M., Giovanoli, C., Scherb, C., Bhat, S. "Agile management in cybersecurity," in Proceedings of Society 5.0 Conference, xth ed. 2023;93:21-32. https://doi.org/10.29007/9fg8.