Adapting Agile Practices to Security Context – Practitioners' Perspective

Katarzyna Łukasiewicz[1(✉)] and Sara Cygańska[2]

[1] Gdańsk University of Technology, ul. Narutowicza 11/12, 80-233 Gdańsk, Poland
katlukas@pg.edu.pl
[2] IHS Markit, ul. Marynarki Polskiej 163, 80-868 Gdańsk, Poland
sara.cyganska@ihsmarkit.com

Abstract. In this paper we explore the problem of introducing agile practices to projects dealing with systems with high security requirements. We also propose an approach based on AgileSafe method and OWASP ASVS guidelines, that could support such introduction. What is more, we present the results of two surveys aimed at analyzing IT practitioners' views on applying agile methods to security reliant systems as well as evaluating the set of agile security-oriented practices which are a part of the proposed approach. This paper is an extended version of the paper "Security-oriented agile approach with AgileSafe and OWASP ASVS" that was published as a part of LASD 2019 conference proceedings [36].

Keywords: Agile · Security · Software development methods

1 Introduction

The concern for providing secure systems has become increasingly important throughout the years. With the rapid progress in the IT domain, expansion of the internet solutions and the level of general computer science knowledge, the problem with security is no longer restricted to government organizations and banking, it involves even small companies that store any private data or engage in the IoT projects.

At the same time, the changing markets and need for flexibility encourages many companies to adopt agile approach [1]. While such approach is known for its benefits concerning effectiveness and client satisfaction [1], when it comes to the security aspect the potential advantages are not that obvious. The core agile methods, such as Scrum [2], eXtreme Programming [3] or Kanban [4] do not mention explicitly security-oriented practices. On the other hand, most of the readily available security frameworks were created with a more disciplined approach in mind.

Taking into consideration the unflagging popularity of agile methods and an increasing concern for security in the IT domain, an approach that would allow to incorporate more security-oriented practices into agile software development would be of value [5].

The goal of the research described in this paper was to identify security-focused agile practices, evaluate their usability and impact so that the positively assessed

A. Przybyłek and M. E. Morales-Trujillo (Eds.): LASD 2019/MIDI 2019, LNBIP 376, pp. 63–81, 2020.
https://doi.org/10.1007/978-3-030-37534-8_4

practices could be incorporated into an OWASP ASVS [6] compliant process, as a part of AgileSafe method [7].

2 Background

2.1 Agile Methods

Ever since the announcement of the Agile Manifesto [8], the agile methods have been growing increasingly in popularity. The reports of the benefits experienced by numerous companies [9, 10] encouraged the trend to shift from traditional, plan-driven methods to the agile ones. What is important, this shift has not only concerned small and evolving companies which are considered a target of the agile approach. Bigger organizations with larger teams or corporate structures have also sought the ways to incorporate agile approach, which resulted in methods such as SAFe [11] or DevOps [12].

2.2 Security Frameworks and Standards

Since the 1990's there have been attempts to formalize guidelines and standards concerning software security. In 1999 ISO [13] proposed Common Criteria for Information Technology Security Evaluation – ISO 15408 [14]. Recognizing the value of unified security standard, governments of Canada, France, Germany, Netherlands, United Kingdom and United States were involved in the creation of this document. The main goal of ISO 15408 was to present formal criteria for security assessment of computer systems. There are other standards that address more specific security concerns such as ISO 27032 [15] and NIST Cybersecurity Framework (NIST CSF) [16] which focus on cybersecurity.

While providing vital information and formal methods to assess security in computer systems, these standards are not directly applicable to agile software development processes. Agile methods are not inherently equipped with security assurance practices. Should such methods be applied in their basic forms, they would struggle to provide conformance evidence for security standards. At the same time, adding traditional security-oriented practices to agile software development process might weigh it down and advantages of the agility could be lost. This opens a room for researching towards a solution which enables to meet the recommendations and to follow the related best practices of secure software development processes while still not backing down from being agile unless it is necessary.

2.3 OWASP ASVS

The name of the OWASP Application Security Verification Standard (OWASP ASVS) comes from the organization with same name, which created it - The Open Web Application Security Project [17]. OWASP is a non-profit organization whose goal is to improve software security. Its mission is defined as improving the visibility of the security problem, both among individuals and organizations, so that they can make decisions on this subject consciously [17]. The organization operates as an open

community in which anyone interested in security issues can participate. It publishes tools and documents that are available under open licenses. Due to the lack of connection with commercial companies, OWASP describes itself as impartial.

OWASP ASVS is directed both to people involved directly in the development process (developers, architects, testers, security experts) as well as their clients. Its two main goals are to help creating and maintaining secure software and help in defining requirements between service providers and their clients. A straightforward language and accessibility make it a practitioners' friendly standard.

The standard distinguishes three levels of requirements for various purposes and the degree of security provided: Level 1: Opportunistic, Level 2: Standard and Level 3: Advanced.

OWASP ASVS has been chosen for this research based on its versatility, open access and popularity among practitioners [18]. The domain of web applications is at the forefront of security issues, with frequent news about major security breaches [19]. For this reason, catering a solution that would allow combining agile security practices with OWASP ASVS requirements could be of interest to many organizations.

2.4 Related Work

Attempts to address the new hybrid approach for security aware agile development are carried out in various ways. One of the ideas is to create new, extended methodologies based on the existing agile ones. An example of this is the Secure Scrum method 20, which extends Scrum. It's been created as an extension to Scrum to support the security assurance. This method presents some valuable practices (such as S-Tag and S-Mark) but focuses only on Scrum and does not address the norms and standards conformance aspect.

Other propositions include frameworks that can be used with any agile methodology - this approach has been used by the aforementioned AgileSafe method but in the safety aspect as well as the method proposed by Veracode. It involves performing Veracode services on user's code to detect vulnerabilities. The service is offered in the cloud and the details of the tests are not visible to the users. Veracode allows security verification to be carried out in several different ways.

The operation of this type of solution is based on the observation that agile methodologies are a set of certain practices. It is possible, therefore, to extend them to new practices, as long as they do not conflict with the existing ones.

3 AgileSafe

The need for hybrid approaches that would allow reconciling regulatory requirements with agile practices is not exclusive to the security context. Safety-critical software development is another, if only more so, highly constrained domain. In the safety context, quite similarly to the security one, norms and standards are vital to ensure the level of trust and quality of high-integrity systems. In order to enable safety-critical software companies to adopt hybrid agile approach while satisfying the regulatory requirements of applicable standards, AgileSafe [21] method has been proposed.

It presents a framework for collecting and suggesting the most suitable agile practices for a given project, as well as the means for managing and monitoring conformance with the applicable regulatory requirements. In this article we present how AgileSafe can be adapted to the security aspect of software development projects and support introduction of hybrid agile approach to such projects while ensuring the compliance with security norms and standards, using OWASP ASVS as an example.

3.1 Overview

As shown in the Fig. 1, as an input to AgileSafe takes the characteristics of a project in which the new approach will be implemented (Project Characteristics) as well as a list of regulations (Regulatory Requirements), which the project needs to comply with.

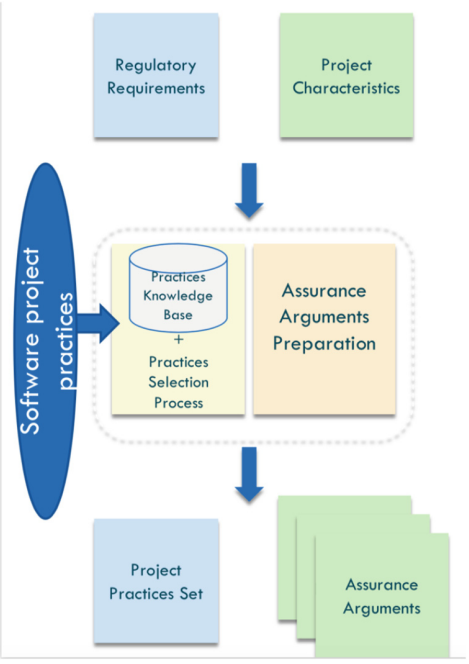

Fig. 1. AgileSafe overview

Based on this information, the user is guided through the process of practices suggestion as well as the process of preparing a set of assurance arguments [22] that will help the user to maintain conformance with given norms and standards. As a result, the user obtains a tailored Project Practices Set, which would suit best a project with given characteristics and regulation restrictions as well as a set of assurance arguments to monitor compliance with the chosen regulations.

3.2 Project Practices Knowledge Base

The information about practices available in AgileSafe, their capability to answer given Project Characteristics and Regulatory Requirements, is kept in the Practices Knowledge Base. Each practice is described using the same template that is then translated into OWL and managed using Protégé tool [23].

An example of a practice description and its capability (type of project for which it works best) for different project factors is presented in the Fig. 2.

At the moment of this research, there were 50 software development practices in the Practices Knowledge Base, including safety-oriented ones.

3.3 Assurance Arguments

In order to ensure that the Regulatory Requirements will be met when applying the new agile approach, AgileSafe uses a set of assurance arguments. The highest level of abstraction is represented by Practices Compliance Argument (and its template Practices Compliance Argument Pattern). It is created separately for each standard added to the method and collects all of the practices from Practices Knowledge Base that have a potential to answer the standard's requirements. Such practices are arranged accordingly in the argument structure for a given standard requirements. Based on these arguments, Project Practices Compliance Arguments are created for each standard that a given project need to comply with, leaving only these practices that will be used in this project. Project Compliance Argument serves as an argument for collecting actual artifacts of the planned practices that serve as evidence in a conformance process. The arguments structure is presented in the Fig. 3.

3.4 AgileSafe in the Security Context

The potential of applying AgileSafe to the projects concerned with security issues has already been presented in [24] based on a case study for clinic appointment/queue management system and IEC 62443-4.1 standard [25]. The promising results of this case study allowed to form another step to further adapt AgileSafe method to cater for security-critical projects and present hybrid security-oriented practices to the Practices Knowledge Base.

4 Security-Oriented Agile Practices

In order to propose agile security practices that could extend the Practices Knowledge Base of the AgileSafe method, a review of the scientific literature and articles on blogs and industry portals was carried out.

4.1 Identification of Security-Oriented Agile Practices

While there are many well-known security-oriented practices such as threat modelling or attack trees, in this research we wanted to expand this list and focus on less obvious, agile inspired practices, to enrich the Practice Knowledge Base of AgileSafe method.

Id	1	
Name	**Abuser stories**	
Description	*Abuser stories are a way of documenting system security requirements. They describe how the system might be attacked and how assets might be put in risk.* *Procedure: Abuser stories are similar to regular user stories - informal and lightweight. They should be written by customers cooperating with developers. The reason is different field of expertise that makes them likely to notice different security issues. Good starting point for writing Abuser stories may be considering system assets. Everything that has value to customer and is accessible by the system might become a target of attack. Another beneficial approach is to try to identify possible attackers, as they characteristic determine nature of attack. Customer industry history is a good resource for such speculations – it contains information that can help identify popular motivations and attack techniques.* *Abuser stories should be assigned a value corresponding with user story scores. Their score should be estimated considering how much damage can be done and probability of successful attack. Abuser story and user story scores should be equal when successful attack described in abuser story devaluates benefits from user story. Assigned scores might be changed as the conditions change (e.g. environment change). Abuser stories should be chosen for sprints to mitigate risks created by developing user stories.*	
Discipline	Architecture	No
	Deployment	No
	Development	Yes
	Environment	No
	Project Management	Yes
	Requirements	Yes
	Test	No
Capability	Factor	Values
	Team Size	A – Under 10 developers; B – From 10 to 50 developers;
	Geographical Distribution	A – Co-located; B – Same building; C – Some working from home; D – Within driving distance; E – Globally distributed
	Domain Complexity	A – Straightforward; B - Predictable; C – Quickly changing; D – Complicated; E – Intricate/Emerging
	Organisational Distribution	A – Collaborative; B – Different teams;
	Technical Complexity	A – Homogenous; B - Multiple technology; C – New technology; D - System/embedded solutions; E – Heterogeneous/Legacy
	Organisational Complexity	A – Flexible, intuitive; B – Flexible, structured; C – Stable, evolutionary;
	Enterprise Discipline	A – Project focus; B – Mostly project focused; C – Balanced; D – Mostly enterprise focused; E – Enterprise focus;

Fig. 2. Abuser Stories practice description

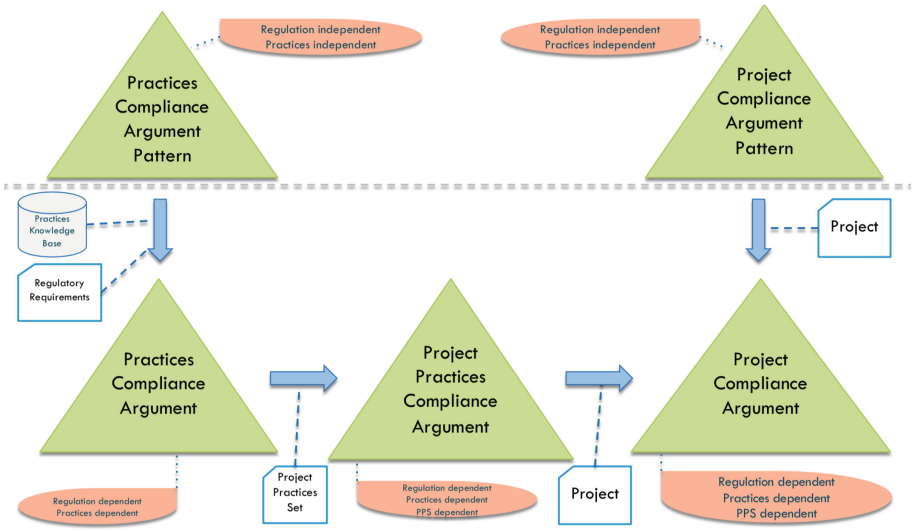

Fig. 3. AgileSafe assurance arguments

The first step to identify the practices was to develop queries that allowed finding suitable sources. The following queries: "secure agile practices", "xp security", "scrum security" were selected through the elimination and trial searches. The search was carried out in the IEEE Xplore digital library, ResearchGate portal and Google Scholar and Google search engines. The initial selection of articles was performed based on the following criteria: titles, summaries (if available) and access to the entire text. The sources were selected based on the fact whether they addressed the subject of security in agile methodologies. The next step was to inspect them for the presence of the agile practices. For further analysis, those articles that described such practices or those that only mentioned them were selected, provided they contained a reference to the source with further information about the practice. A similar selection process was carried out for articles found on the basis of sources obtained in the previous step. As a result, 10 articles were selected to be used in further work [20, 26–28, 30–34].

4.2 Selected Practices Description

Based on the articles identified in the research, 10 hybrid agile security-oriented practices were identified:

Abuser Stories. They describe, using a form similar to regular User Stories, how the system might be attacked and how assets might be put in risk. They should be estimated in accordance to how much damage they may potentially cause and probability of a successful attack [26].

Evil User Stories. This practice describes actions of malicious user (e.g. "As a hacker I want to steal payment information of other clients, so I can sell it."). They may be used as a starting point for threat modelling [27].

Misuse Cases. They are negative use cases. They illustrate behavior not wanted in the system, that can cause a security breach and can be described using UML diagrams [28].

Protection Poker. This is a software security game intended to create a list of each requirement relative security risk. It derives form Planning Poker technique of estimation [29].

Second Delivery. This is a process, that aims to integrate security related solutions to the project that already satisfies functional requirements. It is based on XP methodology [30].

Security Engineer. It calls for adding an expert role, that brings up-to-date security knowledge to developers' team. His insight is useful during multiple phases and actions in project.

Security Sprint. This is a practice inspired by Scrum. It's similar to regular Sprint except that it focuses on security issues [31].

Security-Focused Code Reviews. Such reviews should be performed for every story separately – no story can be completed without security review, fixing findings from review and then passing re-review [32].

S-Mark and S-Tag. Originating from Secure Scrum, they are a way to document identified security issues in Scrum Backlog by creating system of tags (security issues) and markings for stories related to respective tags [20, 34].

Spikes. They are a way to include security analysis and design within Scrum. They accommodate activities that don't produce customer-valued product, like security analysis or system designing [33].

5 Surveys

In order to evaluate the usability and accessibility of the selected security-oriented agile practices in projects with high security requirements two surveys were conducted. The first one (Survey A), analyzing more general security in agile aspects, was focused on gaining information about expected average user awareness of problems and chances related to using agile approach in such projects. The second survey (Survey B) tackled 10 specific agile security-oriented practices, asking the respondents to rate their respective ease of use and security enhancement potential.

Subjects chosen to participate in the surveys were 24 IT practitioners (both development and operations) from 7 different software companies, ranging from small to corporate ones, from Poland and UK. The questionnaire was distributed mostly by emails and direct messages in social networks, eliminating probability of acquiring responses from random, unrelated to the field respondents. The survey was created using Google Forms infrastructure, utilizing both open and closed questions. The respondents were also provided with the practices detailed descriptions.

5.1 Survey A

First three survey questions were used to evaluate participants' experience in agile methods and security related topics. The next two questions assessed benefits and problems related to using agile practices in projects with security requirements. Then respondents were asked about their preferred methodology. All of 24 participants answered those questions.

First question was about years of experience in agile methodologies. The results are presented in the Fig. 4. Almost half of participants declared 1–3 years of experience in agile. Less than ¼ had experience of up to 1 year. Almost 40% had been working with agile methodologies for more than 3 years. Those results suggest that most of the survey participants had at least basic knowledge about agile.

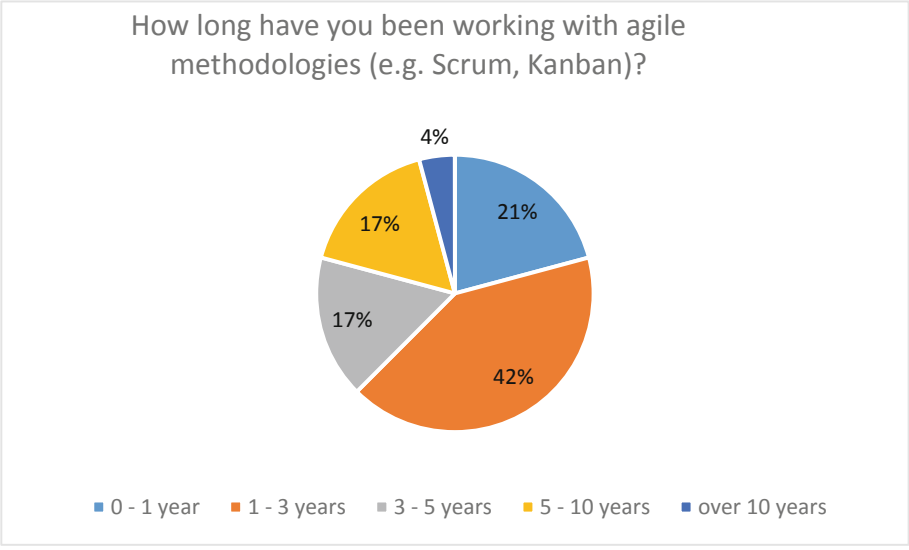

Fig. 4. Experience in agile methodologies

The second question, which results are presented on Fig. 5, was about experience in security reliant projects. Similarly, to the previous questions result, largest group had 1–3 years of experience. ¼ of participants declared experience shorter than 1 year. 17% had more than 3 years of experience, but none had more than 10 years. According to those results participants should have had some knowledge about security related issues.

The third question was about participation in projects with security requirements and utilized methodologies. 75% of respondents participated in such project and out of those 83% used agile methodologies, 11% used traditional methodologies, and 11% worked with different solutions (e.g. hybrid methodologies). For agile practices users, results are shown on Fig. 6. Among agile participants the most popular methodology was Scrum, then Kanban ex-aequo with agile methodologies modification. 13% of

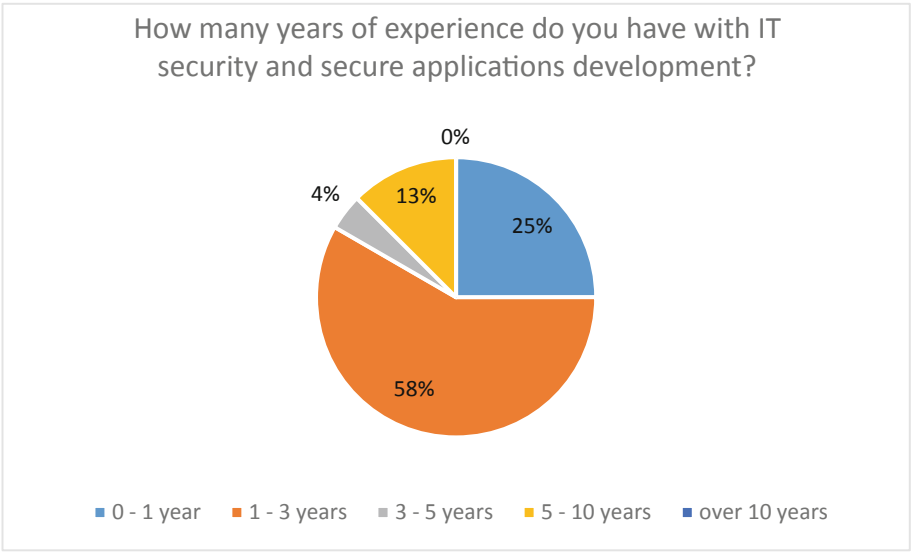

Fig. 5. Experience with security and secure applications development

respondents didn't mention any specific agile methodology. It's worth noting, that none of the respondents who used traditional methodologies had less than 3 years of experience in security.

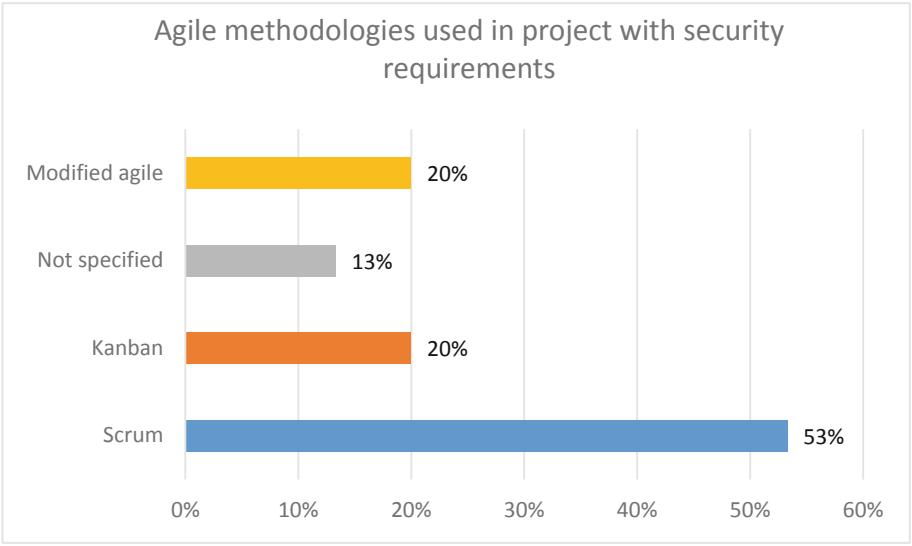

Fig. 6. Agile methodologies used in project with security requirements

The next question was an open one about threats related to use of using agile practices in projects with security requirements. The most popular answers were:

- problems with documentation and conforming to security standards (21%),
- allowing client to prioritize features (17%),
- using User Stories to express security requirements (17%),
- using TDD (due to problems with defining appropriate tests) (17%),
- slowing down development (17%),
- pressure to release product early (13%),
- difficulty to keep some parts of the system secret (13%).

Half of respondents mentioning slow down blamed for it security requirements and only some noticed that it's not dependent on the methodology. Among other mentioned threats were uneven level of knowledge in the team, lot of refactoring and usage of external tools. Also, worth noting is the fact that 17% of the respondents didn't come up with any problem – all of them had less than 3 years of experience in security.

Fourth question was complementary to the third one – it was about the benefits of using agile practices in projects with security requirements. The most popular answers where:

- frequent deployments, pair programming (25%),
- increased speed of development (25),
- increasing security knowledge level in the team – better communication (21%),
- User Stories and Backlog to document security requirements and raise awareness of security issues (21%),
- fast security patches release (17%),
- continuous integration (17%),
- addressing changes in requirements (17%).

Among other answers were also TDD, DDD, preventing errors and rising team awareness. 17% of respondents didn't see the difference in benefits for security and non-security projects and 4% didn't see a possibility to use agile practices in projects with security requirements at all.

Respondents were also asked about their preferred methodology for security related projects. Results are presented on Fig. 7. The vast majority chose agile methodologies, but most did not provide any justification except for curiosity. In that group the most popular was Scrum, then Kanban and their modified versions. The rest didn't mention any specific agile method. 17% of all respondents would use more traditional approach (e.g. V model or Waterfall) due to importance of documentation and planning for security. 75% of them had less than 3 years of experience in both agile and security. 8% would use hybrids of traditional and agile solutions. 4% of participants would match the methodology to project characteristic and the same amount answered that it makes no difference to the developer.

The results of the first survey showed that a lot of projects with security requirements are currently developed using agile methodologies, despite the fact that they are not perfectly suited for the task. Traditional methods created to fit this very purpose are rarely used anymore among the respondents of the survey. Also, most of the participants would like to use agile methodologies for future projects. This shows the need to

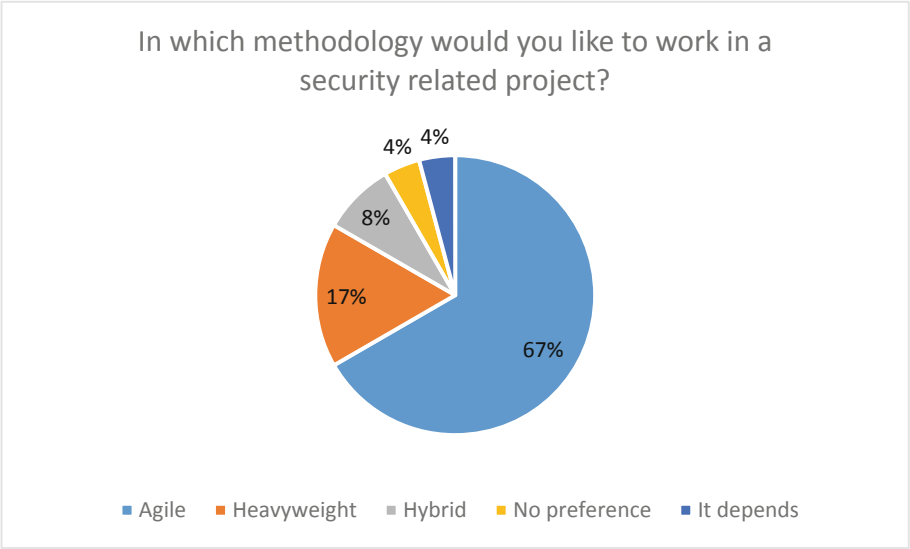

Fig. 7. Methodology choice for security related project

develop new, more agile methods of secure software development. The threats that respondents noticed in agile practices were mostly related to using unmodified versions of those practices, not adjusted to security needs – a place for improvement in this area is clearly visible. Also, it's important to notice that a lot of participants had little or no knowledge of security issues, despite their study and work in IT field. That calls for an effort to popularize the topic among both students and professionals.

5.2 Survey B

For each practice two closed questions were asked about its ease of use and if it's improving security in the project. In total, 15 of the participants made their choices in those questions. Also, each practice was open to comments from the respondents. The results are presented in the Figs. 8 and 9.

Abuser Stories. None of respondents chose negative answer for this practice security improvement potential and not many had doubts about its positive influence. But 27% believed it would not be easy to use - as the reason they mostly described difficulty in estimating attack probability. Despite this fact, this practice has potential benefits in the projects wanting to comply with OWASP.

Evil User Stories. This practice was also positively rated in terms of security improvement. What's more, only 20% expressed doubts or were undecided about its ease of use. Those results categorize it as both efficient and easy to get started with. Respondent commented on possible threat to project agility in case of creating a large number of evil user stories.

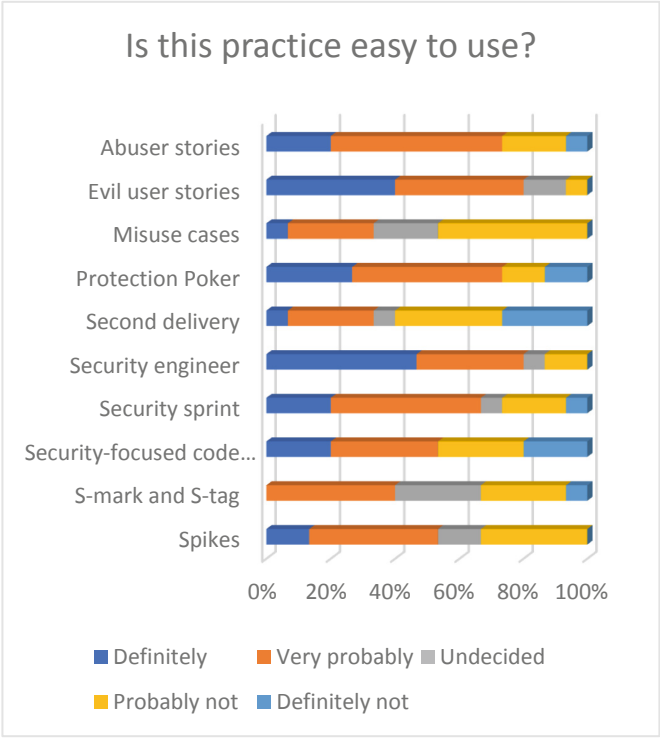

Fig. 8. Practice ease of use

Protection Poker. Majority of respondents found this practice easy to use – among the benefits they listed possible automation of prioritization. The doubts of 27% of surveyed were similar to those for Abuser Stories practice – difficulty in estimation of attack ease and probability. Another noticed difficulty is the necessity for security experts to participate in the process. Despite that problems only 7% didn't rate the practice positively in terms of security.

Second Delivery. This practice didn't occur as easy to use to most respondents (60%). A lot of them were concerned about the need to re-implement huge parts of system in order to satisfy security requirements. 67% of answers in question about security were positive, but considering its difficulty, this practice might not cause some problems in actual development process. Also, a significant problem with security was noticed. During the first development unexpected security flaws might be introduced to the system that are not addressed in the second delivery.

Security Engineer. Most of respondents (80%) rated this practice positively in terms of easiness to use, as it wouldn't require additional amount of work from the team and it would be beneficent to have an expert that is not writing the code himself. Among listed problems were difficulty in finding the suitable person for this role and risk of putting all of responsibility for security on one person. Despite those issues, rating in

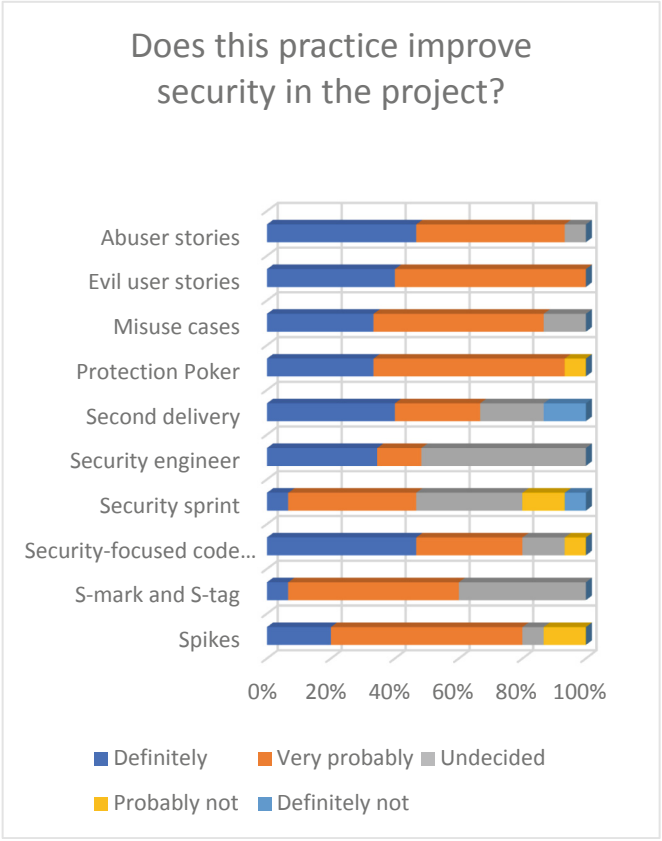

Fig. 9. Practice security improvement

security improvement area was very positive, with only 7% of participant undecided and none rating it negatively.

Security Sprint. The majority of respondents (67%) rated this practice as easy to use, but doubts were expressed that it could lead to development work duplications. Also, the question was asked about the case in which not enough security tasks are defined to fill the whole sprint. 47% of answers were positive in terms of security improvements, but as much as 33% of participants were undecided. This can indicate that practice description should be clarified when added to the Knowledge Base.

Security-Focused Code Reviews. Opinions on this practice's ease of use are divided – the results for definitely and definitely not are equal (20%). Among mentioned problems were difficulty with finding a suitable expert and a lot of additional effort required for conducting such reviews. Despite that, most of respondents decided that this practice improves security in the project (80%). But the expected improvement seems not to be worth the effort required.

S-Marks and S-Tags. None of the respondents found this practice definitely easy to use, and 40% decided it's probably easy to use. Considering amount of answers "Undecided" in both questions, this practice might be too complicated to take up without previous training. Practice gained no negative rating in terms of security, but concerns were raised that it might be possible to lose track of some tags and marks and therefore omit some security issues in development. Also, the question was asked about support in existing project management tools, which could solve tracking problem.

Spikes. Although the majority of respondents (53%) rated this practice as easy to use, 33% doubted it – some commented that it's difficult to understand. However, in terms of security, most of participants expressed no concern about its influence on project security. A question was also asked about other practices that can be used in security projects development. Only two answers were provided – bug bounty and security hackathon. This shows that it's not a common knowledge among developers.

The results show that, although not all practices are easy to use, most of them serve their purpose well by explicitly requiring some security assurance activities. Some of those that scored lowest in terms of easiness might be improved by description clarification, training or providing supporting tools.

6 OWASP Assurance Argument

Because of the positive results of practices security assurance evaluation, the next step was to add them to the Practices Knowledge Base. The selected practices were analyzed according to the AgileSafe practice description template and incorporated into the knowledge base. An example of such description is presented in Fig. 2. Newly added security practices were assessed with respect to their OWASP conformance potential. An example of such assessment is presented in the Fig. 10.

OWASP ASVS requirements has been added to the method and based on the Practices Compliance Assurance Argument Pattern, were mapped to the Practices Compliance Assurance Argument using NOR-STA tool [35]. An excerpt of this argument is presented in the Fig. 11.

All of the OWASP ASVS requirements were successfully mapped into the structure. The practices that were able to answer specific requirements were attached with a relevant rationale in the NOR-STA tool. None of the requirements were left without a practice that might be able to provide conformance.

It is worth noting that there was not one practice that would sufficiently address all of the OWASP ASVS requirements, which means that in a project wishing to comply with the standard, implementing a combination of the analyzed practices would be needed.

The prepared Practices Compliance Argument has been accepted as a part of the AgileSafe potential extension for security assurance domain. Based on this argument, depending on a given project's Project Characteristics, a new hybrid approach with OWASP ASVS compliance potential could be suggested.

Id	1		
Name	**Abuser stories**		
Used in:	Name of the Regulation and regulatory requirement	General Practice	Fact
	OWASP ASVS / 1.10 Verify that there is no sensitive business logic, secret keys or other proprietary information in client-side code.	Abuser stories covering attacks that use sensitive business logic, secret keys or other proprietary information in client-side code can be created.	Abuser stories document security requirements for the project. Those requirements are addressed during development.
	OWASP ASVS / 2.19 Verify there are no default passwords in use for the application framework or any components used by the application (such as "admin/password").	Abuser stories covering attacks that use default passwords can be created.	Abuser stories document security requirements for the project. Those requirements are addressed during development.
	OWASP ASVS / 2.23 Verify that account lockout is divided into soft and hard lock status, and these are not mutually exclusive. If an account is temporarily soft locked out due to a brute force attack, this should not reset the hard lock status.	Abuser stories covering attacks that use lack of hard lock, soft lock and their mutual exclusivity can be created.	Abuser stories document security requirements for the project. Those requirements are addressed during development.
	OWASP ASVS / 2.27 Verify that measures are in place to block the use of commonly chosen passwords and weak passphrases.	Abuser stories covering attacks that use commonly chosen passwords and weak passphrases can be created.	Abuser stories document security requirements for the project. Those requirements are addressed during development.
	OWASP ASVS / 2.31 Verify that if an application allows users to authenticate, they can authenticate using two-factor authentication or other strong authentication, or any similar scheme that provides protection against username + password disclosure.	Abuser stories covering attacks that use username and password disclosure can be created.	Abuser stories document security requirements for the project. Those requirements are addressed during development.

Fig. 10. Abuser stories practice assessment

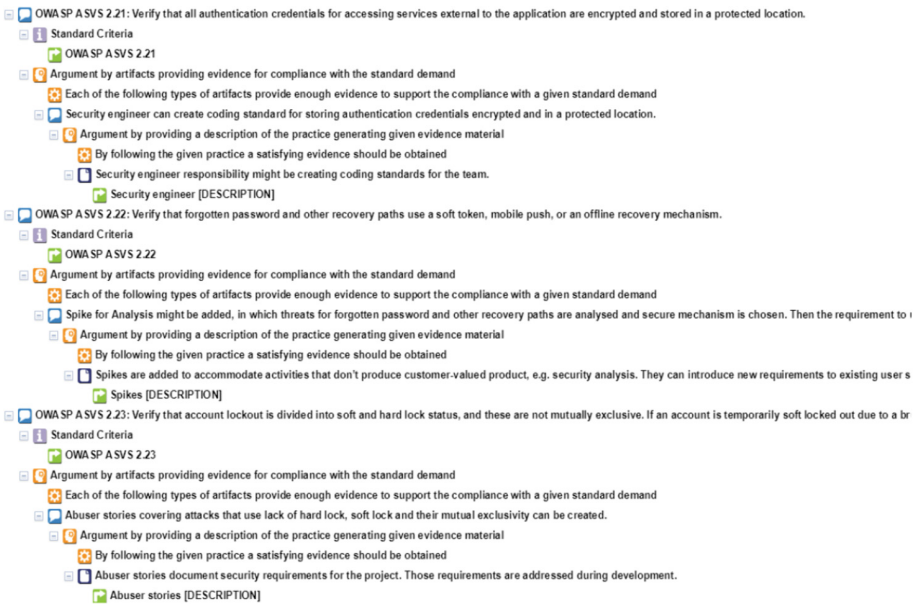

Fig. 11. OWASP ASVS Practices Compliance Argument excerpt

7 Conclusions

During the literature review, 10 security-oriented agile practices were identified. The practices were positively assessed in the conducted surveys and successfully enriched the Agile Practices Knowledge Base. The OWASP ASVS was mapped into the method and formed, along with the identified practices, the Practices Compliance Argument, which after updating it with all of the other applicable practices available in AgileSafe, might be further used to support practices selection in specific projects. A case study carried out with such projects, going through the whole practices selection process of AgileSafe might be performed as next step of the research.

References

1. VersionOne® Releases 11th Annual State of Agile Report (2017). https://www.versionone.com/about/press-releases/versionone-releases-11th-annual-state-of-agile-report/
2. Schwaber, K., Beedle, M.: Agile Software Development with Scrum. Prentice Hall, Upper Saddle River (2002)
3. Beck, K., Andres, C.: Extreme Programming Explained. Addison-Wesley Professional, Boston (2004)
4. Anderson, D.: Kanban. Blue Hole Press, Sequim (2010)
5. Komolo, C., Gomez, M.: Incorporating security best practices into agile teams (2016). https://www.thoughtworks.com/insights/blog/incorporating-security-best-practices-agile-teams

6. Manico, J.: OWASP Application Security Verification Standard (2015)
7. Łukasiewicz, K., Górski, J.: AgileSafe – a method of introducing agile practices into safety-critical software development processes. In: Proceedings of the Federated Conference on Computer Science, vol. 8, pp. 1549–1552 (2016)
8. Agile Manifesto, Manifesto for Agile Software Development (2001). http://agilemanifesto.org
9. Drobka, J., Noftz, D., Raghu, R.: Piloting XP on four mission-critical projects. IEEE Softw. **21**(6), 70–75 (2004)
10. Lindvall, M., et al.: Agile software development in large organizations. Computer **37**(12), 26–34 (2004)
11. Knaster, R., Leffingwell, D.: *S*AFe Distilled: Applying the Scaled Agile Framework for Lean Software and Systems Engineering. Addison-Wesley Professional, New York (2017)
12. Kim, G., Willis, J., Debois, P., Humble, J., Allspaw, J.: The DevOps Handbook. Trade Select (2016)
13. International Organization for Standardization. About ISO - ISO. http://www.iso.org/iso/home/about.htm
14. ISO/IEC, ISO/IEC 15408-1:2009(en) (2009). https://www.iso.org/obp/ui/#iso:std:iso-iec:15408:-1:ed-3:v2:en
15. ISO/IEC, ISO/IEC 27032:2012 (2012)
16. National Institute of Standards and Technology, Framework for Improving Critical Infrastructure Cybersecurity (2014)
17. OWASP. https://www.owasp.org/index.php/Main_Page
18. OWASP users. https://www.owasp.org/index.php/Category:OWASP_Application_Security_Verification_Standard_Project#tab=ASVS_Users
19. World's Biggest Data Breaches & Hacks (2019). https://www.informationisbeautiful.net/visualizations/worlds-biggest-data-breaches-hacks/
20. Mougouei, J.D., Fazlida, N., Sani, M., Almasi, M.M.: S-Scrum: a secure methodology for agile development of web services. World Comput. Sci. Inf. Technol. J. (WCSIT) **3**(1), 15–19 (2013)
21. Łukasiewicz, K.: Method of selecting programming practices for the safety-critical software development projects. Ph.D. dissertation, Department of Software Engineering, Gdańsk University of Technology, Gdańsk, Poland (2019)
22. Górski, J., Jarzębowicz, J., Leszczyna, R., Miler, J., Olszewski, M.: Trust case: justifying trust in an IT solution. Reliab. Eng. Syst. Saf. **89**(1), 33–47 (2005)
23. Musen, M.A.: The Protégé project: a look back and a look forward. AI Matters **1**(4), 4–12 (2015). Association of Computing Machinery Specific Interest Group in Artificial Intelligence
24. Górski, J., Łukasiewicz, K.: Meeting requirements imposed by secure software development standards and still remaining agile. In: Rak, J., Bay, J., Kotenko, I., Popyack, L., Skormin, V., Szczypiorski, K. (eds.) MMM-ACNS 2017. LNCS, vol. 10446, pp. 3–15. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-65127-9_1
25. IEC 62443-4-1 4-1: Secure product development life-cycle requirements
26. Peeters, J.: Agile security requirements engineering. In: Symposium on Requirements Engineering for Information Security (2005)
27. Fischer, E.A.: Federal Laws Relating to Cybersecurity: Overview of Major Issues, Current Laws, and Proposed Legislation (2014)
28. Sindre, G., Opdahl, A.L.: Eliciting security requirements with misuse cases. Requirements Eng. **10**(1), 34–44 (2005)
29. Williams, L., Meneely, A., Shipley, G.: Protection poker: the new software security "game". IEEE Secur. Priv. **3**, 14–20 (2010)

30. Aydal, E.G., Paige, R.F., Chivers, H., Brooke, P.J.: Security planning and refactoring in extreme programming. In: Abrahamsson, P., Marchesi, M., Succi, G. (eds.) XP 2006. LNCS, vol. 4044, pp. 154–163. Springer, Heidelberg (2006). https://doi.org/10.1007/11774129_16
31. Boström, G., Wäyrynen, J., Bodén, M., Beznosov, K., Kruchten, P.: Extending XP practices to support security requirements engineering. In: Proceedings of the 2006 International Workshop on Software Engineering for Secure Systems (SESS 2006). ACM, New York, pp. 11–18. http://dx.doi.org/10.1145/1137627.1137631
32. Nguyen, T.: Integrating Security into Agile Methodologies. http://www.umsl.edu/~sauterv/analysis/F2015/Integrating%20Security%20into%20Agile%20methodologies.html.htm
33. OWASP: Agile Software Development: Don't Forget EVIL User Stories. https://www.owasp.org/index.php/Agile_Software_Development:_Don%27t_Forget_EVIL_User_Stories
34. Pohl, C., Hof, H.-J.: Secure Scrum: development of secure software with scrum. In: Securware 2015: The Ninth International Conference on Emerging Security Information, Systems and Technologies (2015)
35. NOR-STA project Portal (2017). www.nor-sta.eu
36. Łukasiewicz, K., Cygańska, S.: Security-oriented agile approach with AgileSafe and OWASP ASVS. In: Proceedings of the 2019 Federated Conference on Computer Science and Information Systems, Leipzig, Germany (2019)