Journal of Theoretical and Applied Information Technology
31st December 2019. Vol.97. No 24

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

# NEW HYBRID METHODOLOGY FOR SECURE SYSTEMS

**[1]MOHAMMAD ZAKARIA, [2]ABDALLAH QUSEF, [3]ISRAA ALBADARNEH, [4]AALAA ALBADARNEH,**

[1]Student, Princess Sumaya University for Technology, Department of Software Engineering, Amman, Jordan

[2]Assistant Professor, Princess Sumaya University for Technology, Department of Software Engineering, Amman, Jordan

[3]Student, University of Jordan, King Abdallah II School of Information Technology, Amman, Jordan

[4]Student, Princess Sumaya University for Technology, Department of Computer Science, Amman, Jordan

E-mail: [1]mohdzak89@gmail.com, [2]a.qusef@psut.edu.jo, [3]eng.i.abdullah25@gmail.com, [4]eng.aalaabadarneh@gmail.com,

**ABSTRACT**

Software security is considered as a time consuming and after-thought activity, this could be because of its complexity or the lack of enough knowledge. Today`s software development companies are focusing on high speed software delivery, however, many companies are still using the traditional methodologies. In this paper, we propose a new secure methodology named NHMSS, which aims to give security its needed attention by the development team, the proposed methodology mixed Waterfall and Scrum to produce a hybrid software development approach that will apply the rules of both of them. The methodology will integrate security activities according to well-known security processes and best-practices.

**Keywords:** *Software development methodology, Security, Secure System, Agile, Waterfall, Scrum.*

## 1. INTRODUCTION

Software security is a significant and evolving problem (Ayalew & Kidanem, 2012) that is not given it`s required attention during software development (AlAzzazi & ElSheikh, 2007), which turn out to be worse later on because of the increased complexity and extensibility in new software products. The need for secure systems is increasing and becoming an important factor of software development. According to Symantec`s internet security threat report statistics. Figure 1 (Symantec, 2015) presents the total number of vulnerabilities discovered only after they are exploited by the attackers.

In order to protect a system against harm, attention must be given to its requirements, similar to other system properties and quality attributes (Romero-Mariona et al, 2009). Most of the software development companies are considering Security as an after-thought (Alnatheer et al., 2010; Futcher & Solms, 2008), they might look at some common points such as securing passwords, data encryption and other minor security activities on early phases of software development, then, security might not be taken into consideration anymore.

Software development methodologies are evolving now, some companies are using the traditional Waterfall approach, others are using Agile Scrum or Agile Extreme Programming (XP), and others might use a mixed approach. Each methodology has its own characteristics, advantages and disadvantages, there are no perfect methodology that fits all projects. Traditional approaches are characterized as plan-driven, slow, and rigid approaches, Agile approaches are well-known with their flexibility, change accessibility and short development increments.

In recent years, there are some numerous studies, such as: (Baca & Carlsson, 2011; Tetmeyer, 2013; Alnatheer et al., 2010; Romero-Mariona et al., 2008; Mead & Stehney, 2005; Flechais et al., 2004; Romero-Mariona et al., 2009) that were focusing on integrating security into the software development process. To satisfy the needs for building secure software, there is a need to develop a software development approach that could guarantee security at each phase of the software life cycle. In this study, a new hybrid secure software development

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

methodology has been presented, the proposed methodology will mix Waterfall and Scrum approaches, and integrates some of the most compatible Security Engineering (SE) activities to allow the development team to consider security in all phases of the development life cycle.

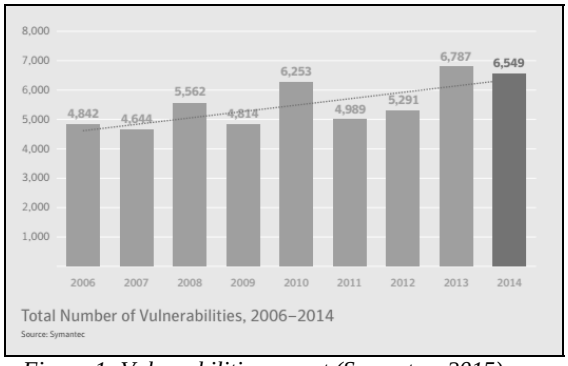


*Figure 1: Vulnerabilities report (Symantec, 2015)*

The need for building secure software is increasing, and software development companies are not giving security the required attention, especially at early development life cycle phases, which are the most important phases to have security in mind, and since software grows up through its life cycle, software development methodologies should give special attention to security aspect of the product (Ayalew & Kidanem, 2012). The main motivation of this paper is to integrate security engineering activities into the software development life cycle, there is no way to apply all the available security standards and best practices on the development process as they take a large amount of time and efforts. Several methodologies had been created, and each one of them focuses on one or two aspects, without focusing on the overall security of the software.

The new proposed methodology will combine the best features of the traditional software development and Agile software development to come up with a new hybrid approach that aims to have the features of both and build secure software. Security will be integrated into the new methodology by adding the best well-known and compatible security activities from different security standards, process and best practices. It has been proven that using a methodology to write a code increases the quality of the code produced (Grembi, 2008).

Currently, software security is a serious problem and may become much worse in future due to the increase in complexity, connectivity and extensibility; so, there is a need to develop an approach for software development that could guarantee security at each phase of software life cycle (Ayalew & Kidanem, 2012).

Security activities could be performed according to the software development approach; each security activity has its own compatibility with the phases of software development. Some security engineering activities are only compatible with traditional approach and some of them are only compatible with Agile, therefore, it is hard to stick to one approach while applying security at the same time, and that`s why a hybrid approach is proposed on this paper. Security activities had been chosen after a deep look at the main and well-known security engineering processes and identifying what specific security activities they perform.

Vulnerabilities in software's are evolving, and cyber-attacks are becoming crucial. Security is not considered in the development life cycle of the software, mainly because more attention is given to other activities, or because of the lack of security knowledge. Applying security activities into the current software development methodologies could be difficult and could affect the development process and require more efforts.

The problem concerns on how to consider security in all the development life cycle phases while maintaining the strength of the selected development approach.

This paper is organized as follows, section two presents related work. Section three introduces and discusses the proposed framework, the evaluation results are presented in section four, and finally, section five is a summary of this paper, including the conclusions and suggested several ideas for related future work.

## 2. RELATED WORK

The researchers looked on the topics of information security and software development methodologies, some of them had issues that were relevant to the topics of implementing security activities into traditional and Agile development methodologies, where other papers identify, analyzes and compares different security standards, procedures and best practices. Some other papers introduce and describes Waterfall and Agile approaches.

Hossein Keramati, et al (Keramati, 2008) research was about identifying and evaluating security practices and activities in SDL and CLASP. It represents an algorithm, called agility degree that was used to rate security activities and its compatibility with Agile approach. Mikko Sipnon, et al. (Sipnon, et al, 2005) identified the problems with integrating security in Agile, in

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

addition to creating a new Agile friendly method that implement and track security features. Annette Tetmeyer (Tetmeyer, 2013) proposed an approach for eliciting, analyzing, prioritizing and developing security requirements. Gustav Bostorm, et al. (Bostrom, et al, 2006) proposed a new way to extend extreme programming practices to support security requirements engineering. Konstantin Beznosov and Philippe Kruchten (Beznosov & kurchten, 2005) examined how conventional security assurance suits Agile methodologies for developing software-intensive systems. Ahmed Alnatheer, et al. (Alnatheer, et al, 2010) presents and describes an empirical study on the effects of using security issues with Agile methodologies. Dejan Barca and Carlsson (Baca & Carlsson, 2011) proposed a security enhanced development process which use security activities from already established security engineering processes. Ayalew and Kidanem (Ayalew & Kidanem, 2012) research introduce a way to bridge the gap in Agile model and security by providing insightful understanding of the SE process that are used in the current Agile industry. Fangkun Yang (Yang, 2013) research explores ways to migrate from Waterfall developing approach to Agile approach. Juyun Cho (Cho, 2009) proposed a new hybrid software development method for large-scale projects, also this research analyzes characteristics, strengths, and weaknesses of both conventional and Agile methods. Nayan B. Ruparelia (Ruparelia, 2010) identifies and analyzes difference software development lifecycle models. Most of these studies were focusing on integrating security activities into Agile; as these activities are most compatible with traditional development approaches, in addition, they have not integrated these activities on all the development life cycle phases.

Susan M. Mitchel (Mitchel, et al, 2009) introduces a systematic review that compares Waterfall and iterative development methodologies. CLASP and SDL had been evaluated and compared by Johan Gregorie (Gregorie, et al, 2007). A detailed comparison between Agile and traditional software development methodologies has been presented by M.A. Awad (Awad, 2005). A comparative study on Agile software development methodologies has been introduced by A B M Moniruzzaman (Moniruzzaman & Hossain, 2013). Zornitza Bakalova and Daneva (Bakalova & Daneva, 2011) describes his case study research on client's participation in a traditional and in an Agile software company. Another Agile and traditional development methodologies comparison has been made by Irit Hadar on his research (Hadar & Sherman, 2012). These comparisons were very useful for the researcher to understand the main characteristics of both development approaches.

Lynn Futcher (Futcher & Solms, 2008) provides guidance to software designers and developers by defining a set of guidelines for secure software development. Michael Kainerstorfer (Kainerstorfer, et al, 2011) describes the experiences and lessons learned by using SDL for the development of a small real world software project. John Biega (Viega, 2005) describes how to build security requirements in a structured manner that is conductive to iterative refinement. Asim El Sheikh (AlAzzazi & ElSheikh, 2007) presents a process for the development of security critical software projects and an overview of some of the existing processes, standards, life cycle models.

Gottipalla, A. K., (2013) shows that although there are many operations and methodologies which may support secure software development, very few methods are designed specifically to face up software security completely. Security engineering activities was integrated into the agile software development process (ben Othmane, L., 2014) this research extended the agile software development process with secure development activities, at the end of each iteration this method enabled ensuring the delivery of secure software. (Oueslati, H., 2016) study 20 challenges that reported in 28 publications of developing secure software using the agile approach, it was founded that the challenges are related to the software development life-cycle, incremental development, security assurance, awareness and collaboration, and security management. The researchers illustrate the validity, limitations, and impacts of the challenges.

Microsoft present a process named SDL (Howard, 2006) which integrated different security activities into the development life cycle phases, Microsoft also introduces SDL-Agile to apply SDL activities on Agile development methodology (Howard, 2006). OWASP introduces a process to develop more secure software under the name CLASP (OWASP, 2015), it provides a set of security activities to be integrated into software development life cycle phases. SSE-CMM framework (SSE-CMM, 2007), and Common Criteria (Common Criteria, 2007) represent methods to evaluate security engineering processes and activities.

Vähä-Sipilä, A. (2013) proposes a risk-based approach for developing secure software. The method is based on managing the product security

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

risks and implementing security solutions. The Characteristics of next-generation security was addressed by (Dove, R. 2009). Next-generation security is an arising property of the system and involved six characteristics as self-organizing, adapting to unpredictable situations, and harmonious with system purpose. (Dove, R., 2016) Presented an overview article which studied a wide range of the systems engineering community, it was concluded that with an agile-attack environment, agile system-security is necessary.

Security development models that used to secure web application were investigated in (Shuaibu et. al, 2015), through 499 publication agile development models seem to have more attention due to the multiple stakeholders that are discussing the security viewpoints. Geogre Grispos and other researchers (Grispos, et al, 2015) presented the Security Incident Response Criteria (SIRC) which can be used to evaluate existing security incident response solutions, and can be applied to a various security incident response approaches. Willett and other researchers (Willett, et al, 2015) applied the fundamentals of agile systems engineering to show the application of cybersecurity decision pattern (CDPs) and the cybersecurity decision pattern languages (CDPL) in the design of agile cybersecurity operations.

Rindell, et al. (Rindell, et al, 2015) used an evaluation framework to show the requirement for security assurance of agile methods, combining with Microsoft SDL security framework, against Finland's' established national security regulation (Vahti). Mohd Nazir supported the idea that agile methodologies seems to be best for security development of small software, to implement a security in agile: developing an iterative approach with check points in order to analyze the effectiveness of security capabilities. The study of Steve Harrison (Harrison, et al, 2016) examined the software development approaches within the United Kingdom Government, this study utilize the Open Web Application Security Project (OWASP) to allow security and agile working together.

Because of the increasing rate of software security vulnerabilities, some efforts have been started to create secure development methodologies and security standards, these attempts are mentioned shortly in the following Table:

Most of the related studies were focusing on a specific security activity or a specific software development methodology, others were comparing security activities or evaluating them. The main contribution of this research is to propose a new secure software development methodology, the proposed methodology will not describe how to build fully secure software, and it aims to consider security activities in all phases of the software development life cycle. Security activities have been chosen from different well-known security processes and best practices, these activities will be achieved using two different development approaches; Waterfall and Scrum.

## 3. THE PROPOSED METHODOLOGY

This section will introduce the proposed methodology, New Hybrid Methodology for Secure Systems (NHMSS). This methodology is a software development methodology for creating secure software; securing the software will be based on using different security activities that will be integrated during all phases of the development life cycle. NHMSS combines two main software development approaches; traditional Waterfall and Agile Scrum, it will combine and use the best features of them and will come up with a new hybrid approach.

Many researchers have found that security is not considered at early phases of development, in addition, they argued that most of the available security activities are not compatible with fast-driven methodologies and they slow down the development process, as they have been build according to traditional heavy-weight approaches. The proposed methodology will try to overcome these obstacles and will introduce a new way to build secure, traceable, and documented software. In particular, developing a secure software system is a complex and time-consuming process that seeks to accommodate frequently competing factors, such as functionality, scalability, simplicity, time-to-market (Flechais, et al, 2007). It has been said that "Information security should be an integral part of the development process and should be taken into account at every stage of the SDLC" (Futcher & Solms, 2008). NHMSS aims to introduce security to the development community as a basic activity, it does not ensure that security issues will not occur, but it will reduce the chance of certain types of security issues to happen. An overview of NHMSS is shown in Figure 2.

NHMSS is defined abstractly as follows:

- High-level requirements, design and analysis phases will be followed as in traditional Waterfall approach.
- User requirements, development, testing and deployment phases will be followed as in Agile approach.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

- Security activities will be performed during all these phases, mainly on early phases, and they are all extracted from different security engineering processes, standard and best practices.

More details will be demonstrated in the following subsections.

### 3.1 Combining Waterfall with Agile (Hybrid)

There are some aspects of software development project can benefit from an Agile approach and others can benefit from a more predictive traditional approach (Awad, 2005). NHMSS combines Waterfall traditional approach with Agile Scrum approach.

The first phases (requirements, analysis and design) will be carried out using traditional Waterfall approach, whereas the other phases (development, testing and maintenance) will be performed using Agile Scrum. Unfortunately, the demand of quick release and constant flow of new products has forced companies to move away from methodical and Big Design Up Front (BDUF) Waterfall process which includes most SE processes (Baca, 2012). NHMSS will meld traditional and Agile approaches together in a way that maintains the principles of both; i.e. the team doesn't have to stick to traditional or Agile approach. The proposed methodology will try to oblige the strong points while suppressing the shortcomings of both approaches and use a Hybrid approach. At the end, there is no "one-size-fits-all" solution (Awad, 2005), so going Hybrid might satisfy all the needs.

### 3.2 Security Advisor

The proposed methodology focuses on integrating security activities into the software development process, a security expert or security knowledge is needed in the development team, thus, two scenarios are proposed:

**1)** Having a dedicated security engineer/advisor in the team who is responsible for security engineering activities and any security related tasks, in addition to providing the team with proper security awareness (education) and support the development process. Having a security advisor is a common security activity in both SDL and CLASP processes (Howard, 2006), (OWASP, 2015).

**2)** If the team does not have a dedicated security engineer/advisor; the project manager should enroll some developers into some security courses in order to improve their security knowledge, for example: secure coding skills, secure design, in addition to any courses/certificates that could help them understanding main security concepts and the most common attacks. OWASP has published a cheat sheet that contains a large number of vulnerabilities and how to mitigate them (OWASP, 2015). The team could have one or two developers only with a good security knowledge/experience that could help the team applying security activities. Having a security expert close by is advantageous to the team and, more importantly, to the customer (Davis, 2006). Also, the existence of the security engineer could help spread the tacit knowledge of security rather than having to do the same through documentation with the same effect (Alnatheer, et al, 2010). In addition, Alnatheer claimed that Agile team needs a security engineer onboard.

### 3.3 Traditional Approach

NHMSS will start with the well-known traditional Waterfall approach, which is based on strict planning, detailed requirements elicitation, expansive design, and heavy documentation. The traditional Waterfall approach will be used for the following reasons:

1) To benefit from its advantages; mainly requirements tracing and documentation.
2) Providing the team time needed to digest the application properties (requirements, database definition and goals).
3) To perform security activities at early phases of software development. Security activities demand a good planning and detailed documentation, which is very suitable to traditional approach.
4) To better understand the system and have a strong starting point for the team.
5) Waterfall is widely used and preferable by many software development companies.

#### 3.3.1 Security and Traditional Approach

As mentioned earlier, most of the available security processes are based on the traditional development approach as they are somehow rigid activities. Waterfall model tries to address security issues before software is released (Ayalew & Kidanem, 2012), because of the detailed planning and requirement elicitation. The proposed

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

methodology starts with traditional approach (which is very suitable for security activities to be performed) by defining security requirements and performing other early phases security activities.

### 3.3.2 High-Level Requirements

Traditional approach is well-known for planning out a large part of the software process in comprehensive details for a long time. Clear and written requirements will help the team in planning and decision making. These requirements are the base of the software; they will be considered as abstract, specific and high-level requirements that are necessary to build the system, they are very essential and they could be branched to another sub-requirement(s) later after moving to Agile`s Scrum product backlog. These requirements will be elicited by the stakeholder and analyzed by the development team. They will be considered as a (vision) high-level requirements, which are well-known by some development teams as a "scope", while the other detailed requirements will be elicited in Agile product backlog section. The team will decide how important is each requirement and everything will be discussed with the stakeholder.

Several studies argued about that security must be considered at early development phases such as: (Ayalew & Kidanem, 2012; Alnatheer, et al., 2010; AlAzzazi & ElSheikh, 2007; Ruparelia, 2010) and others. The team might not have a clear idea what kind of security they really want at the first glance, but they can define the main abstract requirements, and add any other requirements later on as explained.

Figure 3 shows that any requirement could be branched to sub-requirements when moving to Agile.

There are two main kinds of requirements, functional and non-functional. Functional requirements are concerned with software services such as the scope and the required data structure for the software, it specifies what the system should do. Non-functional requirements are concerned with software constraints such as performance, reliability, usability and security, it describes how the system works and behaves.

Current development practices suffer from different key problems like, security requirements tend to be kept separate from other system requirements, and not integrated into any overall strategy (AlAzzazi & ElSheikh, 2007). Also, many studies have found that security activities are based on traditional approach and not compatible with Agile such as: (Ruparelia, 2010), (Ayalew & Kidanem, 2012; Keramati, 2008; Alnatheer et al., 2010; Bostrom et al., 2006) and others. NHMSS will integrate security at the early traditional phases and will suggest some security activities on other Agile phases.

Security requirements needs to be adequate as possible, they need to be precise, complete, explicit, and non-conflicting with other requirements; this could be accomplished by having dedicated requirements for security and by considering them from the beginning.

### 3.3.3 Design and Analysis Phases

The main objective of these phases is to determine requirements, understand and analyze them to develop software that addresses these requirements. This could be done by examining each requirement in details, analyzing and studying the system and characterizing user requirements. These phases need powerful social, communication and administrative skills to be performed. The design phase mainly will identify all inputs, outputs and the needed process, In addition to full database creation. The team will start analyzing each requirement, creating database tables, and outlining the main screens. The analysis phases answers any inquiries of who will use the software, what it will do, when and where. The team will identify improvement opportunities, and develop ideas for the new software. The team will also start writing the required software documentation, documents such as: (BRS, SRS, High-level design, SDS, and functional design). "For the security analysis and security design part of the process, it is important to ensure that expert knowledge is available in order to identify threats and countermeasures (AlAzzazi & ElSheikh, 2007). At this stage identify the security threats and their potential impact, then for each threat direct a mitigation strategy. Different type of threat can be occurred, such as spoof the user's identity. In addition an attacker can perform a denial of service attack against the system software. Attacker can obtain sensitive information like user's password (Matthee, 2014).

The team will perform the following security activities during these phases:

1) **Document the security architecture of the software.**

The team should write a document that specify the security architecture of the software as the following example:

1) User access mechanisms.
2) The integrity of the communication.
3) Encryption and Decryption.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

4) Digital signatures and authorization mechanisms.

2) **Create Abuse cases**

Use cases are well-known for the development team in defining an interaction between an actor and the system, abuses cases are not very different, it just looks into the system by the eye of the illegitimate user, abuse cases could be very useful in increasing both customer and user understanding of the security features of the software, they are also very useful to define the security requirements, it only needs some brainstorming from the team to demonstrate abuse cases. (Bostrom, et al, 2006) recommends the importance of having a security engineer while defining abuse cases; a fluent up-to-date knowledge of security vulnerabilities is needed for threat identification.

**3) Analyze the attack surface**

Attack surface analysis is about mapping out what parts of a system need to be reviewed and tested for security vulnerabilities. The point of attack surface analysis is to make developers and security specialists aware of what parts of the application are open to attacks and to find ways to minimizing them (OWASP, 2015). This activity could be done by a security engineer or by the developers.

**4) Security Education**

SDL stated that "Acquiring security knowledge could be as simple as reading appropriate sections in a book or watching an online training class" (Davis, 2006). According to the identified security requirements, the team should have some security sessions, or at least, selected members from the team must have security certification. Futcher & Solms (Futcher & Solms, 2008) mentioned that building secure software begins with the effective education of software development teams.

**5) Other activities**

Many activities could be performed through the development life cycle of the project. There are a large number of security-related activities that are all useful and could produce very secure software; risk management activity as an example. This activity is widely used, it can prevent many kinds of attacks, but it takes a large amount of time and effort, and it requires an expert knowledge, therefore, it will not be used on this proposed methodology, it will be optional to use according to the security engineer or project manager. Risk management normally requires a considerable amount of time, effort and expertise to obtain optimum results (Futcher & Solms, 2008). Threat modeling is a very important activity as well, and it used as an essential activity on SDL and CLASP.

#### 3.3.4 Documentation in Traditional Approach

Software documentation is one of the main reasons why the proposed methodology started with traditional Waterfall approach. It is a highly important activity in Waterfall approach, it is a rigid activity that requires the team to document everything, unlike Agile which demand a "working software over comprehensive documentation" (Agile Alliance, 2005). Documentation is also considered as an advantage when a new member joins the team at the middle of the project, if there were some written documents that explains the project scope and other details, all he/she needs to do is to read these documents and start working as fast and easy as possible. Documentation is also very important in regard to security, as "lack of documentation is viewed by security experts as a lack of compliance proof resulting in delays and possible rejection in submission for operational certification" (Woody, 2013). Different studies have mentioned the importance of documentation in software development process (Awad, 2005; Wayrynen, et al., 2004; Winston, 1970).

### 3.4 Agile Approach

The Second part of the proposed methodology will be followed according to Agile approach; each of the development, testing, maintenance and deployment phases will be achieved in Agile. Agile is based on iterations, or sprints; each iteration adds business value to the product, Agile is also a flexible, adaptive, and collaborative approach and brings innovation to the team.

In this part, the flexibility and freedom of Agile starts, teams can adapt changing requirements easily under the abstract requirements that Waterfall defined earlier. This part might not adhere to Agile manifesto, but it is build depending on Agile approach, which indicates communication between team members instead of heavy documentation. Security experts often criticize Agile for having a fundamental lack of an inherent security mechanism to produce secure software (Alnatheer et al., 2010). The proposed methodology will use some security activities extracted from well-known security processes and standards, it will suggest some Agile-compatible security activities based on literature studies that had found the most compatible security activities with Agile.

Agile approach will be used for the following reasons:

1) To speed up the development process.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

2) To improve customer collaboration.
3) To add/edit requirements on Scrum product backlog.
4) To use Agile-compatible security activities.

It also has to be mentioned that "Agile alone cannot be responsible for solving security issues for a given piece of software in development" (Alnatheer et al., 2010).

### 3.4.1 Security and Agile Approach

A key benefit of integrating security into the Agile methodology is the relative increase in security awareness in designers and developers, and managers of the project because in the normal Agile process developers and architects are not usually concerned with security issues (Alnatheer et al., 2010). As mentioned earlier, security and Agile are barely compatible, however, some activities have been proposed while maintaining the agility of the software. After performing the Waterfall traditional activities, NHMSS will use Scrum as an Agile method. Scrum is aimed at providing an Agile approach for managing software projects while increasing the probability of successful development of software (Moniruzzaman & Hossain, 2013).

Scrum has a special flavor in team member's roles and responsibilities, the team will be re-constructed and this can affect the success of the software development. Scrum also has dedicated meetings where the collaboration and innovation happens. These meetings could be very helpful for security integration, for example, having a security engineer among the team could spread the security knowledge by brining security issues to the forefront of the team discussions, and he/she also could attach security recommendations and assess the security implications of security requirements. The team could extend the iteration planning meetings to have some time for security issues.

### 3.4.2 Secure Software Development

Development phase is very important in Scrum; each software functionality will be developed according to sprint/feature goals. Everything is fully-defined and clear from the beginning as NHMMS starts with a traditional Waterfall approach which defines everything at the beginning; this will be considered as a big advantage to the Scrum team. As been mentioned earlier, security and Agile are not compatible, NHMSS focus on security activities that could be performed on early traditional phases, meanwhile, some security activities have been found compatible with Agile, in addition to some light security activities that could be performed are suggested.

The following are some security activities that could be performed during Agile part:

• Adhering to Coding standards: Coding standards should be shared with the developers by the security engineer. These standards could guide the developers in writing secure code. "By adhering to coding standards, developers can prevent the introduction of many flaws that can lead to security vulnerabilities" (Futcher & Solms, 2008).
• Pair Programming: Pair programming is having a security engineer next to the developer shoulder. Pair Programming provide an additional security argument for better assurance (Alnatheer, et al., 2010) and could help in writing secure code. Even a programmer's security experience level plays a major role in the development of secure software.
• Code Analysis: NHMSS suggests using code analysis activity to improve the security of the written code, this could be done using manual or automated techniques, noting that static code analysis is a common security activity in SDL and touch points process Also, (Keramati,2008) has found that static code analysis activity is compatible with Agile. Figure 4 demonstrated an overview of the security activities performed by NHMSS.

### 3.4.3 Testing Phase

Scrum is about producing a shippable software increment at the end of each sprint. After each sprint, the development team will do all the testing themselves, and then it can be handled to Quality Assurance (QA) for detailed testing. There is no separate testing phase; the test should be performed during the entire sprint.

Quality Assurance and business analysts can prepare the test cases during the development time, these test cases could be very beneficial to address vulnerabilities and security issues, and once all cases are completed and accepted the team will move to the next sprint. Security testing could be performed here, as mentioned earlier, code analysis is a very important security activity, and it is considered as a testing-phase activity. Security testing should be considered during software testing; both SDL and CLASP put a lot of focus on security testing (Futcher & Solms, 2008). In addition, "Security testing by contrast should bring

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

security assurance to the customer. Integrating security early on helps with integration because continues integration is part of the standards model of behavior in Agile development" (Alnatheer, et al., 2010).

Penetration testing could be performed here, but it is not an easy task and it could take time, cost and effort. But it has to be mentioned that, test results are not enough to ensure that a system is secure, tests can only show that the system passed the test, not that it is safe against future attacks (Wayrynen, et al., 2004). Fuzz testing is an easy activity that also could improve the security assurance, it could be performed by sending random data to the software as an input and see what the software will produce, it could help the team finding implementation bugs.

#### 3.4.4 Sprints

Sprints is what made Agile special, the team can deliver software features in increments. What can be really beneficial to security here is having a sprint that dedicated only to security, the team will have a chance to solve the security issues and figure out how to write better, more secure code.

#### 3.4.5 Maintenance and Deployment Phases

These phases will be performed as any methodology, no specific security activities are required here, and however the team can integrate any security activities from the available standards and best practices.

#### 3.4.6 Customer Collaboration

Customer collaboration is one of the reasons why the proposed methodology use Agile after starting with Waterfall. The time has changed, and technologies are all over the world, most of people are technology educated and have a good or little technology knowledge; everything around us are somehow is using technology, therefore, customer collaboration is demanded now days in software development, their involvement is important for the success of the software, "Since the customer can see the process move ahead frequently, they would much more content that what is being done is worthwhile and they can also provide valuable feedback for better productivity and security of the final delivery" (Alnatheer, et al., 2010).

#### 3.4.7 Documentation in Agile

Documentation reduces agility, that`s what we have been hearing all the time, but as explained before in the proposed methodology, documentation will be performed in the early Waterfall part, where the most important things needs to be documented.

In Agile part, especially in product backlog, where requirements are added to the software, documentation will be minimized or will be set to an optional activity, however, it is recommended to keep documenting everything during the software lifecycle. In Scrum, everyone is responsible for documentation, not only the QA`s and Business Analysts (BA), and that`s could be very beneficial for future needs, for example, the developer doesn't need to spend a long time understanding and remembering the logic that had been written a year before. The priority will always be to sprint deadline, if the team did not finish the documentation, it can be scheduled later. So, NHMSS recommends writing the most important documentation rather than not writing any documents at all. It should be a part of continuous delivery.

To illustrate further about how to use NHMSS, a written case study is provided in Appendix A.

## 4. RESULTS AND DISCUSSION

As real-world experimentation is difficult and costly, an interview-based evaluation with practitioners' method has been chosen to evaluate the proposed methodology NHMSST. The evaluation is done by conducting interviews with well-positioned and experienced project managers and software development professionals from different software development companies. The evaluation process was done by asking them open-ended questions and having their feedback on this study.

### 4.1 Interview Process

Seven intensive interviews have been conducted in order to evaluate this study; all the participants were from international private software development companies. They are all located in Jordan, but their customers are distributed around the Middle East, Europe and Africa. The interviewees had different roles in software development such as senior consultants, Team

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

leaders and project managers, their experience varied between nine and sixteen years.

First step for interviews was getting an agreement of having this interview, then scheduling a time. After that, each interview started by introducing the researcher experience in software development and information security, and how he the information has been collected, then introducing the proposed methodology to the interviewee, explaining each phase and each activity. After that a written cases study (Appendix A) explained and discussed; which has been written as a simulation to real-world experimentation. Then they have been asked sixteen open-ended questions. All interviews were face-to-face interviews, only one was done using online video call because of the global location. Each interview took between forty-five to ninety minutes to complete.

Interview questions are categorized as follows:

1) Questions related to software development methodologies.
2) Questions related to security in software development.
3) Questions related to the using of Hybrid approach.

Interview questions and answers are shown in Appendix B.

### 4.2 Interview Results

After analyzing the interviews, the following points are a summary from their answers:

• Software documentation is very important and it's highly demanded by software development professionals.

• The currently used security activities in software development companies are very basic and old-fashion, even the security knowledge of theses software development professionals is very specific and outdated.

• Traditional Waterfall approach is still used until now by various software development companies, and it will be always needed for some projects, however, other professionals thinks that there is no need for the proposed approach nowadays.

• It is possible to migrate from traditional to Agile, but it needs commitment and the ability to change from the team. In addition, software development professionals recommend having a middle ground between Agile and traditional Waterfall for software development.

• All of the interviewees thinks that security activities could slow down the development process, moreover; they haven't used or tried them before.

• At the same time, most of them thinks that using the proposed methodology will build secure software and they are willing to see a real-world experimentation of it.

• The interviewed professionals think that the proposed methodology retains the dependency tracking and clarity of traditional development whilst embracing the strength and innovation of Agile development at the same time, however, only one of them argued that applying it could be costly.

### 4.3 Comparison with Previous Works

In this subsection, comparison between the overall results and previous studies in proposing secure software development methodologies is presented in Table 2. The main contributions of previous studies were integrating security activities in a specific development approach or a specific phase of the development life cycle, where the proposed methodology describes a new way to consider security in the phases of software development life cycle.

## 5. CONCLUSION AND FUTURE QORK

This research proposed a new hybrid methodology to build secure software's, it is based on two software development approaches; traditional Waterfall, and Agile Scrum. The two approaches have been combined in order to ensure a structured planning, flexibility, speed and compatibility. There is no one approach that fits to all projects, in addition, it has been found that security activities are not compatible with all the development approaches; as some of them was created using traditional development in mind and they are only compatible with traditional approaches, where some of these activities are Agile compatible.

The proposed methodology has been introduced to seven highly qualified software development professionals to evaluate it and to have their feedback on it, they all approved that it could build secure software, while most of the negotiations was about using the hybrid approach as they are convinced that using one approach might be a better solution.

As future work, more deep analysis is needed to discuss the advantages and disadvantages of using a hybrid approach, in addition to a measurement of the proposed security activities and their ability to build secure software. Moreover, Scrum has been

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

chosen as the Agile approach in the proposed methodology, there are a variety of Agile approaches that could be integrated while using a hybrid approach.

**REFRENCES:**

[1] AlAzzazi, A. & ElSheikh, A. (2007). Security Software Engineering: Do it the right way.Arab Academy for Banking and Financial Sciences, Amman, Jordan.

[2] Alliance, A. (2001). Agile manifesto. Retrieved September, 2015, from http://www. Agilemanifesto. Org.

[3] Alnatheer, A. & Gravell, A. & Argles, D. (2010). Agile Security Issues: A Research Study.University of Southampton.Southampton, UK.

[4] Awad, M. (2005). A Comparison between Agile and Traditional Software Development Methodologies. The University of Western Australia, Crawley, Australia.

[5] Ayalew, T. & Kidanem, T. (2012). Identification and Evaluation of Security Activities in Agile Projects.Master`s thesis, Blekinge Institute of Technology, Karlskrona, Sweden.

[6] Baca, D (2012).Developing Secure Software in Agile Process. Doctoral Disseration, Blekinge Institute of Technology, Sweden.

[7] Baca, D. & Carlsson, B. (2011).Agile Development with security engineering activities. ICSSP'11,149-158.

[8] Bakalova, Z. & Daneva, M. (2011).A Comparative case study on clients participation in a 'traditional' and in Agile software company. Profes2011, 74-80.New York, USA.

[9] ben Othmane, L., Angin, P., Weffers, H., & Bhargava, B. (2014). Extending the agile development process to develop acceptably secure software. IEEE Transactions on Dependable and Secure Computing, 11(6), 497-509.

[10] Beznosov, K. & Kruchten, P. (2005). Towards Agile Security Assurance.University of Britich Columbia, Vancouver, Canada.

[11] Bostrom, G. & Wayrynen, J. & Boden, M. (2006). Extending XP practices to support security requirements engineering. SESS'06, 11-17, Shanghai, China.

[12] Cho, J. (2009). A Hybrid Software Development Method for Large-Scale Projects: Rational Unified Process with Scrum.Issues in Information System, X (2), 340-348.

[13] Davis, N (2006). Secure Software Development Life Cycle Processes. Retrieved June,2015 from https://buildsecurityin.us-cert.gov/articles/knowledge/sdlc-process/secure-software-development-life-cycle-processes

[14] Dove, R. (2009). Embedding Agile Security in Systems Architecture. Insight, 12(2), 14-17.

[15] Dove, R. (2016). AGILE SYSTEM-SECURITY: SUSTAINABLE SYSTEMS EVOLVE WITH THEIR ENVIRONMENT. Insight, 19(2), 8-12.

[16] Flechais, I. & Mascolo, C. & Sasse, M. (2007).Integrating Security and Usability Into the requirements and design process. International Journal of Electronic Security and Digital Forensics, 1(1), 12-26, Geneva, Switzerland.

[17] Flechais, I. & Sasse, M. & Hailes, S. (2004).Bringing Security Home: A process for developing secure and usable systems. New Security Paradigms Workshop 2003, 49-57, Ascona, Switzerland.

[18] Futcher, L. & Solms, R. (2008).Guidelines for secure software development.SAICSIT2008, 56-65, Wildrness, South Africa.

[19] Gottipalla, A. K., Desai, N. M. S., & Reddy, M. S. (2013). Software Development Life Cycle Processes with Secure. The International Journal of Scientific and Research Publications, 3, 1-3.

[20] Gregorie,J.& Buyens, K.& Win, B.& Scandarito, R.& Joosen,W.(2007).On the secure software development process: CLASP and SDL compared.SESS07,Leuven,Belium.

[21] Grembi, J (2008). Secure Software Development. USA, 25 Thomson Place Boston, MA: Cengage learning.

[22] Grispos, G., Glisson, W. B., & Storer, T. (2015). Security Incident Response Criteria: A Practitioner's Perspective. arXiv preprint arXiv:1508.02526.

[23] Hadar, I, & Sherman, S (2012).Agile vs.Plan-Driven Perceptions of Software Architecture. CHASE2012 Conference, 50-55.Zurich.Switzerland.

[24] Harrison, S., Tzounis, A., Maglaras, L. A., Siewe, F., Smith, R., & Janicke, H. (2016). A Security Evaluation Framework for UK E-Goverment Services Agile Software Development. arXiv preprint arXiv:1604.02368.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

[25] Howard, M. Lipner, S., "The Security Development Lifecycle - SDL: A process for Developing Demonstrably More secure Software", Microsoft Press, 2006.

[26] Kainerstorfer, M. & Sametinger, J. & Wiesauer, A. (2011).Software Security for Small Development Teams - A Case Study. iiWAS2011, 305-310, Ho Chi Minh City, Vietnam.

[27] Keramati, H. & Hisseinabadi, S. (2008). Integrating Software Development Security Activities with Agile Methodologies.Sharif University of Technology, Tehran.

[28] Matthee, M. H. (2014). SECURE SOFTWARE DEVELOPMENT FRAMEWORK: PRINCIPLES AND PRACTICES. SANS Institute. Retrieved from https://www.sans.org/reading-room/whitepapers/securecode/agile-defensive-perimiters-forming-security-test-regression-pack-35617.

[29] Mead, N. & Stehney, T. (2005).Security Requirements Engineering (SQUARE) Methodology. SESS'05.1-7, MO, USA.

[30] Moniruzzaman, A. & Hossain, S. (2013).Comparative Study on Agile software development methodologies. Global Journal of Computer Science and Technology, 13(7).

[31] Nazir, M. (2015). Agile Model of Software Security: Risk Perspective. International Journal, 5(11).

[32] Oueslati, H., Rahman, M. M., ben Othmane, L., Ghani, I., & Arbain, A. F. B. (2016). Evaluation of the challenges of developing secure software using the agile approach. International Journal of Secure Software Engineering (IJSSE), 7(1), 17-37.

[33] OWASP, Comprehensive, lightweight application security process, http://www.owasp.org, 2015.

[34] Rindell, K., Hyrynsalmi, S., & Leppänen, V. (2015, June). A comparison of security assurance support of agile software development methods. In Proceedings of the 16th International Conference on Computer Systems and Technologies (pp. 61-68). ACM.

[35] Romero-Mariona, J. & Ziv, H. & Richardson, D. & Bystritsky. (2009).Towards Usable Security Requirements. CSIIRW'09, Article No. 64.

[36] Romero-Mariona, J. & Ziv, H. & Richardson, D. (2008).SRRS: A Recommendation System for Security Requirements. RSSE'2008, 50-52, Georgia, USA.

[37] Ruparelia, N. (2010).Software Development Lifecycle Models. ACM SIGISOFT Software Engineering Notes,35(3), 8-13

[38] Shuaibu, B. M., Norwawi, N. M., Selamat, M. H., & Al-Alwani, A. (2015). Systematic review of web application security development model. Artificial Intelligence Review, 43(2), 259-276.

[39] Sipnon, M. & Baskerville, R.s& Kuivalainen, T. (2005).Integrating Security Into Agile Development Methods.Proc of the 38th Hawaii International Conference on System Science.

[40] Sonia, & Singhal, A. & Banati, H. (2014).FISA-XP: An Agile-based Integration of Security Activities with Extreme Programming. ACM SIGSIFT Software Engineering Notes, 39(3), 1-14, Delhi, India.

[41] Tetmeyer, A. (2013). A POS Tagging Approach to Capture Security Requirements within and Agile software Development process.Master`s Thesis, University of Kansas, Kansas, USA.

[42] Vähä-Sipilä, A. (2013). Product security risk management in agile product management.

[43] Viega, J. (2005).Building Security Requirements with CLASP.SESS'05, 1-7, MO, USA.

[44] Wayrynen, J, & Boden, M, & Bostrom (2004). Security engineering and extreme programming an impossible marriage. Extreme Programming and Agile Methods - XP/Agile Universe 2004.117-128.

[45] Willett, K. D., Dove, R., & Blackburn, M. (2015, October). Adaptive Knowledge Encoding for Agile Cybersecurity Operations. In INCOSE International Symposium (Vol. 25, No. 1, pp. 770-792).

[46] Woody, C (2013). Agile security-review of current research and pilot usage. Software Engineering Institute, Carnegie Mellon University.

[47] Yang, F. (2013). How to Migrate from Waterfall development approach to Agile approach. Master's Thesis, Chalmers University of Technology, Gothenburg, Sweden.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

*Table 1: Related work*

| Paper Title | Authors | Publication Year | Idea and Contribution |
|---|---|---|---|
| A process for developing secure and usable systems (AEGIS) | Flechais et al. (Flechais, et al, 2004) | 2004 | Developing secure systems. It is based on risk analysis, asset modeling, security requirements identification and usability context. |
| Security Quality Requirements Engineering (SQUARE) Methodology | Nancy R. Mead et al. Software Engineering Institutes Networks Systems Survivability (NSS) program (Mead & Stehney, 2005) | 2005 | A methodology for security requirements elicitation only. |
| SRRS: a recommendation system for security requirements | Jose Romero-Mariona et al. (Romero-Mariona, et al, 2009) | 2008 | Recommends the most appropriate security requirements elicitation approach for specific project characteristics. |
| Towards Usable Cyber Security Requirements – SURE | Jose Romero-Mariona et al. | 2009 | A new approach called SURE (Secure and Usable Requirements Engineering), it increases the usability in security requirements specifications by supporting the derivation of testing artifacts from them. |
| FISA-XP: An Agile-based Integration of Security Activities with Extreme Programming | Sonia et al. (Sonia, et al, 2014) | 2014 | A methodology for developing secure systems. It integrates security activities with core Extreme programming based on their degree of agility. |

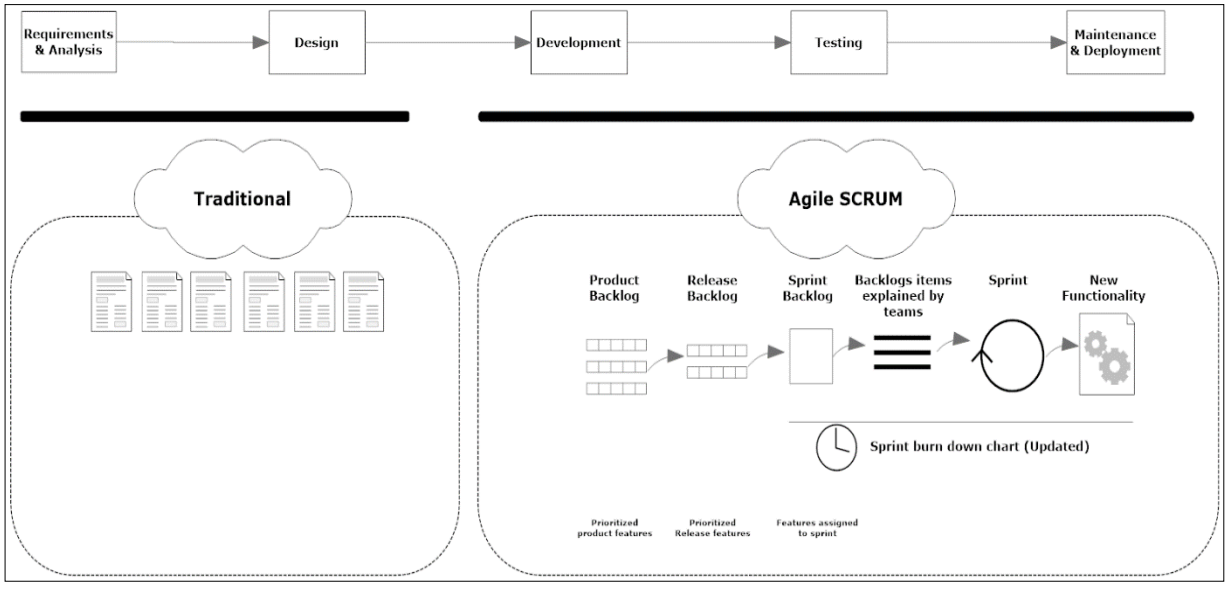


*Figure 2: NHMSS*

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

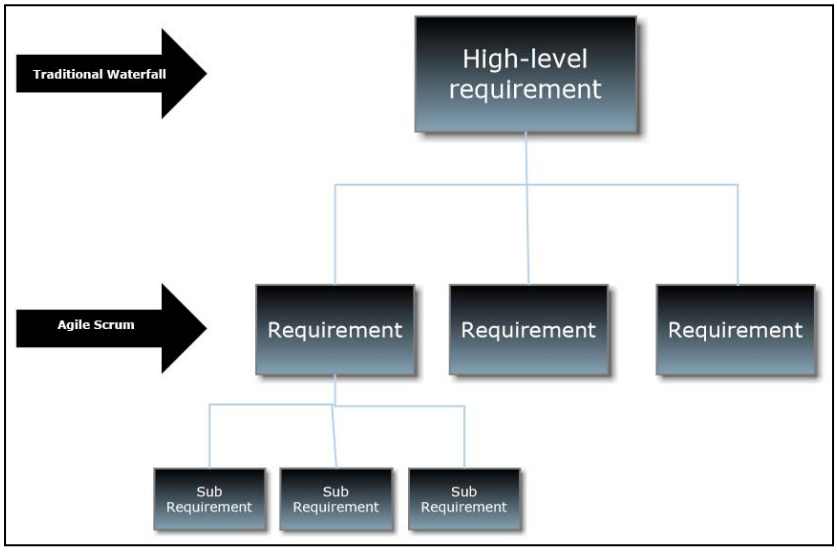


*Figure 3: The Requirements in NHMSS*

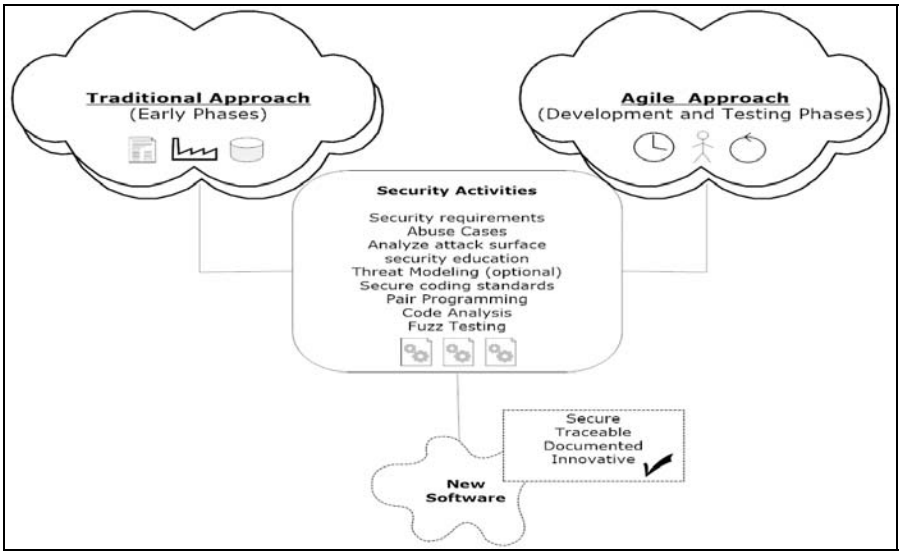


*Figure 4: NHMSS Security Activities*

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

*Table 2: Comparison with Previous Works*

| Paper Title | Authors | Idea and main contribution | Differences with the proposed methodology |
|---|---|---|---|
| A process for developing secure and usable systems (AEGIS) | Flechais et al. | A methodology for developing secure systems. It is based on risk analysis, asset modeling, security requirements identification and usability context. | This study is based on building secure software according to risk analysis and asset modeling activities which are considered as comprehensive and time consuming activities, and that could force the development team to stick with traditional development approach only. In addition, the methodology is not considering security in all the development life cycle phases.<br><br>While the proposed methodology is considering light-weight security activities in all the development phases. |
| Security Quality Requirements Engineering (SQUARE) Methodology | Nancy R. Mead et al.<br><br>Software Engineering Institutes Networks Systems Survivability<br><br>(NSS) program | A methodology for security requirements elicitation only. | This study focuses on the requirements phase only, while the proposed methodology is taking security into consideration in all the development life cycle phases. |
| SRRS: a recommendation system for security requirements | Jose Romero-Mariona et al. | Recommends the most appropriate security requirements elicitation approach for specific project characteristics. | This methodology focuses on the requirements phase only, while the proposed methodology is taking security into consideration in all the development life cycle phases. |
| Towards Usable Cyber Security Requirements | Jose Romero-Mariona et al. | A new approach called SURE (Secure and Usable Requirements Engineering), it increases the usability if security requirements specifications by supporting the derivation of testing artifacts from them. | This methodology focuses on the requirements phase only, while the proposed methodology is taking security into consideration in all the development life cycle phases. |
| FISA-XP: An Agile-based Integration of Security Activities with Extreme Programming | Sonia et al. | A methodology for developing secure systems. It integrates security activities with core Extreme programming based on their degree of agility. | This study is proposing a secure development methodology by integrating security into Extreme Programming approach, which is considered an Agile development approach, while the proposed methodology is using a hybrid approach by starting with traditional Waterfall and moving to Agile Scrum in the middle of the development life cycle. That will satisfy both, Waterfall and Agile advocates. |

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

# Appendix A – Case Study

ABC Tech is a middle size software development (proposed) company that is mainly specialized in web development, they have multiple web development departments, and each department is specialized in a specific type of products/programming languages.
One of these departments consists of:

- A department manager.
- A development team leader, who is responsible for all the development process, in addition to a full supervision on his team members.
- Six developers (two senior developers, three middle experienced developers and one junior developer); they are specialized in web and database development.
- Two quality assurance employees.
- One Business analyst.
- One designer.
- One deployment and support engineer.
- One Information security engineer.

The department manager, the team leader, the business analyst, in addition to a sales representative from the company had a meeting with a new customer (a bookshop owner) who is planning to create a website for his bookshop. The main reason for this initial meeting is to get the main requirements and understand what he needs.
The bookshop owner wants to build a web application that will be used by his employees and his customers; the employees will use the website to register member's information, such as contact and billing information, in addition to book lending information. The customers will use the website to view books in stock and lend or buy books. The bookshop owner mentioned that his customer's information must be maintained in a very secure manner. He also explained in details how he imagines the web application would be, he mentioned that he would like to be always updated and have a look at every release, also he mentioned that it doesn't matter to have the first releases with minimum features and adding more features in the next releases.

The team leader wrote all the Main (High-level) requirements, and informed the customer that the first release could take some time while all the other releases will be available in a short time. The team could start on a very rigid step-by-step process. The detailed start-up process will give the team members some time to get to know one another's styles and strengths. It also provides some time to digest the application requirements and database table definitions.
The team leader will be using NHMSS (New Hybrid Methodology for Secure Systems) as a software development methodology for this project, he need to consider security requirements in his planning and requirements elicitation process, these requirements are defined by the customer or suggested by the security engineer, for example:

- How many failed login attempts the user could have.
- How the passwords are going to be encrypted in database; are they going to be hashed and salted to protect against rainbow attacks?
- How the logs will be written to support forensics.
- How the transactions integrity will be maintained.
- The availability of the system.
- Continuous requirements, for example, validate all HTTP post parameter values against valid character whitelists.
- Implementing three factor authentications.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

The team leader had a meeting with the QA, BA and the senior developers; they demonstrate the main screens and the functionality of the web application in an abstract view as explained by the bookshop owner. The QA and BA started to write the needed documents (BRS, SRS, High-level design, functional design) These documents will be written, maintained and updated for the team to understand the work accordingly.

They will also create use cases and design the main screens in collaboration with the designer. Meanwhile, the team leader (or the security engineer) will review the needed security activities that needs to be implemented, and according to that, he might ask for some security education sessions for the developers if needed. In order to ensure that no requirements are missed out; a traceability matrix could also be designed and maintained.

According to NHMSS the team will switch now to Agile Scrum development, all the written documents are already distributed to the whole team to understand the project and have an overview about it. An already written development standards documents will be sent to developers as well. This standards document also contains secure coding standards that must be followed to write a secure code. Standards make products easier to build because they give direction, and because they are written down as plans and formulas.

The team will be organized to form a Scrum team; they will use an appropriate management/leadership style to maximize motivation and empowerment. The product backlog will be created by transforming the main requirements to it and add some sub-requirements accordingly (when needed). Once all the requirements are listed and agreed by the customer, the next task is to prioritize them in order to pick them up in sprints.

The product backlog requirements are prioritized queue of business and technical functionality, it is possible to add/edit any requirement as it is all Agile in here, but the team cannot go back to the main high-level requirement as they have been elicited during the traditional Waterfall requirements phase; this is not a big issue since any new requirements could be connected to a one of the high-level requirements as a subset of it.

The team will identify the requirements and decides what needs to be done for the first release. These requirements (user stories) will goes under the release backlog, then they will be prioritized with estimation for each item, the estimation will be in hours or days. Some features might take some time to be completed.

As most of the researchers concluded; Most of the security activities are not compatible with Agile. But fortunately, they have found that static code analysis is the most compatible security activity that could be implemented in Agile. The researchers proposed the following security activities (In addition to Static code analysis) that could be implemented without slowing the development process:

- **Pair Programming**

  Perform a real time security reviews of the system design and code.

- **Fuzzy Testing**

  A software testing technique used to discover coding errors and security loopholes by inputting massive amounts of random data, called fuzz, to the system in an attempt to make it crash. (Reference will be added).

- **A dedicated sprint for Security review**

  Sprints will be planned; each sprint could take from two to thirty days. The release backlog will be spitted into sprints backlogs (subset of release backlog). At the end of each sprint, the team will have a fully tested product with all the features of that sprint. QA will test the product in increments by testing the requirements covered in each sprint. But after all the sprints are complete and the final build is ready and all integrated, the QA should test the complete system and should perform end-to-end testing. This should be done in a completely new environment. The burn down chart will be used to monitor the progress of each sprint. It provide day by day measurement of the amount of work that remains in a given sprint or release. The team will have a daily Scrum meeting to follow up with everything and stay synced. The documentation during these phases will be optional, it will be adopted when it is needed only. Finally, the team will have sprint retrospective meeting where they check what went right, what went wrong and the areas of improvements. According to Scrum, the team will have the following main meetings:

- **Sprint Planning Meeting**

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Negotiate which product backlog items they will attempt to convert to working product during the sprint

- **Daily Scrum Meeting**
  The Scrum development team members spend a total of fifteen minutes reporting to each other. Each team member summarizes what he did the previous day, what he will do today, and what impediments he faces.

- **Sprint Review Meeting**
  After sprint execution, the team holds a sprint review meeting to demonstrate a working product increment to the product owner and everyone else who is interested.

- **Sprint Retrospective Meeting**
  The team reflects on its own process. They inspect their behavior and take action to adapt it for future sprints.

The bookshop owner will be involved after every release to take his notes and see how to improve the system and add more functionality if needed. Sprints will be performed until the software is fully created and deployed.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

# Appendix B – Interview questions

1. What is your role and responsibility in development team?
2. How many years of experience do you have?
3. What experience do you have from of working with software development with a plan-driven (traditional Waterfall) approach?
4. What experience do you have from of working with software development with Agile approach?
5. Have you ever considered any security activities in your development process? If not please mention why, if yes please mention what kind of activities you have been using?
6. Is there still a need for traditional Waterfall approach? Why?
7. Have you experienced migrating from plan-driven approach to Agile? What is the main difficulty to do that?
8. What about having a middle ground between Agile and traditional Waterfall for companies that have a balance of control and agility?
9. Do you think that documentation is important? What kinds of documents are necessary?
10. Is there a difference in the project manager`s responsibility in Waterfall development approach as compared to the Agile approach, for instance in choosing team members, and quality assurance assistance? Please explain.
11. Do you think that my proposed methodology retains the dependency tracking and clarity of traditional development whilst embracing the strength and innovation of Agile development at the same time? Please explain.
12. Do you think that my proposed methodology provides the flexibility and transparency that are necessary to adapt the fast changing requirements of stakeholders? Please explain.
13. Do you think that the security activities that my methodology proposed are enough to build secure software?
14. Do you think that these security activities could slow down the development process? Why?
15. Please define what do you mean by “Secure Software”?
16. According to your experience and after reading the case study example, and because your thoughts will be considered as an evaluation for my thesis, do you think that the proposed methodology is applicable? Please explain, and please tell me your feedback regarding every phase of my proposed methodology.