Improvement Model of the Security of Agile Software Process Model

Md. Selim Hossain

Lecturer, Department of Computer Science and Engineering
Khwaja Yunus Ali University, Sirajgonj
E-mail: selimtee@gmail.com
Web: selimtee.orgfree.com

Imtiaz Ahmed

Department of Information & Communication Technology,
Mawlana Bhashani Science and Technology University,
Santosh, Tangail-1902, Bangladesh
Email: imtiazahmed12033@gmail.com

Abstract: Agile is one of the most popular methods for software improvement and its foundation is from its declaration published by a group of software practitioners in 2001.The focus of agile is on developers and customers with the objective to produce working software quickly and accept changes throughout its lifecycle. Agile development methodologies are gaining acceptance in the software industry. If they are to be recycled for constructing security-critical solutions, what do we do about guarantee? This paper scrutinizes how conventional security assurance suits agile methodology for increasing software-intensive systems. It classifies security assurance methods and techniques with regards to their clash with agile development.

Keywords: agile software development (ASD), security, risk analysis, secure software.

Biographical notes: Md. Selim Hossain is a Lecturer in Department of Computer Science and Engineering at Khwaja Yunus Ali University. He has completed his B.Sc. degree on ECE from Hajee Mohammad Danesh Science and Technology University, Dinajpur, Bangladesh. and Pursuing his M.Sc. degree on ICT at Mawlana Bhashani Science and Technology University, Santosh, Tangail-1902, Bangladesh. His main research interest is on the area of Software Engineering, Database Management System, Web development, Cryptography and Network Security

Imtiaz Ahmed is a MSC student in the department of Information and Communication Technology at Mawalan Bhashani Science and Technology University, Santosh, Tangail-1902, Bangladesh. He obtained his Bachelor degree in of Information and Communication Technology from Mawalan Bhashani Science and Technology University, research interest is in the area of Cryptography, Network Security and Medium Access Control Protocol (MAC-Protocol).

1. Introduction

Agile software development designates a method to software improvement under which supplies and explanations evolve through the collaborative effort of self-organizing cross-functional teams and their customers or terminal users. It advocates adaptive planning, evolutionary development, early delivery, and continuous improvement and it inspires rapid and flexible response to transformation. Agile software improvement is based on the following principles; Customer contentment by delivering early and continuous of valuable software; Comfortable changing requirements even in late development; Working software is delivered frequently that will be Close, daily cooperation between business people and developers and Projects are built around motivated individuals who should be confidential to the Face-to-face conversation is the best form of communication. We can say that software is the primary amount of progress for sustainable improvement that enable to Maintain a constant pace for continuous attention to technical and good design. In further way Simplicity the art of maximizing the amount of work that the Best architectures, requirements, and designs emerge from self-organizing teams and regularly the team reflects on how to become more effective adjustment. In software development, agile methods have grown more and more conventional. They are also gradually more used in the improvement of web and network applications where security risks are prominent characteristics. In spite of these risks, most existing agile development methods have few unambiguous features that specifically address security. As a result, security is often added afterwards or included in the process by way of external resources [1]. The agile processes often impose

limitations on the development projects [2]. For instance, it is no longer possible to create a complete picture of a product as all requirements are not yet known and no attempts are made to acquire this information [1][9]. In traditional software development, security managers that perform audits or acts as a security advisor to the developers usually handle all matters related to security. However, in an agile development process, where iterations are short and changes made by the minute, it is not always possible to include a security manager from the outside. There are several challenges that limit the use of ASD for developing secure software such as lack of complete view of the system, absence of security engineering activities in the development process, lack of detailed documentation, lack of security awareness of the customers, and conflict of interests between security professionals and developers[11][12]. The paper is organized as follows. First, we provide an overview of the agile software development approach, then discuss about the security concerning matters and finally proposed security mechanism of agile software improvement.

2. Background

This section gives an overview of the agile software developments (ASD) approach, secure software development and security assurance cases of agile software process model. There are several encounters that perimeter the use of agile software for developing secure software for example lack of complete view of the system, absence of security engineering events in the development process lack of detailed citations, lack of security awareness of the customers and clash of interests between security professionals and developers [15], [16], [17]. Several solutions have been proposed for extending the ASD process to produce secure software but they either fall short of ensuring the security of the increments produced in each iteration, or require performing a long list of security verification and validation tasks that verification requirements [18]) in each iteration which implies that all security requirements must be implemented in the first development reiteration. This paper aims to extend the agile development process to produce acceptably secure software in each phase.

2.1 Agile Software Developments

The ASD approach is specified in the manifesto of four values: individuals and interactions, working software, customer cooperation, and responding to change of the era. The approach is implemented by several methods including: Scrum [5], extreme programming (XP) and agile modeling [6]. These processes are quite different from each other; each has its special central features And appeals to different types of projects, but they desire a certain number of common characteristics. They are all fundamentally iterative. They do not replicate the traditional linear sequence of requirements, design, implementation, and test, rather they repeat this sequence again and again, exploiting the fact that software is extremely soft, modifiable, and has no associated manufacturing cost. The traditional waterfall lifecycle includes some feedback loops, some refinements, as developers cannot get everything right in one pass, but generally rework is considered a bad thing that should be minimized at all costs. The agile methods refers to a group of software development models that is dynamic, adapted to the specific circumstances based on the incremental and iterative method in which the increases are small and normally new releases of the system are created and made available to consumers every limited weeks [7]. It can be recycled with any type of the project but it essentials more engagement from the consumer and to be interactive [4].

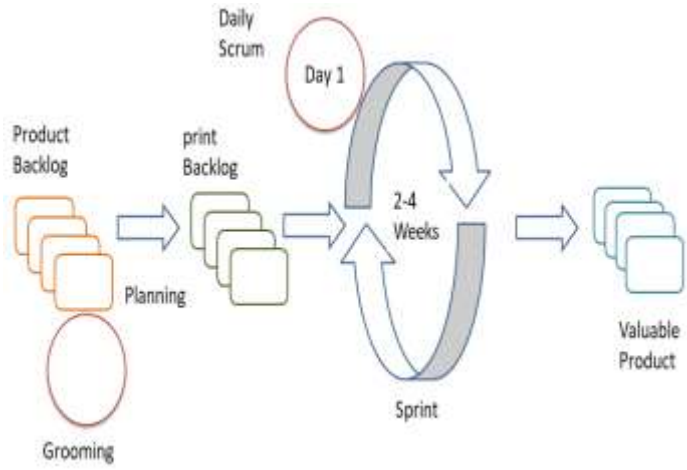

Fig. 1: Agile Software Process Model [19]

2.2 Secure Software Developments

Secure software are developed using processes that integrate security activities for capturing and refining protection requirements and for ensuring their integration into the software through purposeful security design [7]. A known reference model of engineering secure software is the System Security Engineering Capability Maturity model. The process has three sub processes: risk process, engineering process, and assurance process. The risk process enables identifying threats to and vulnerabilities of a given system along with their associated impacts and likelihood of occurrence [6]; that is, their risks. The engineering process supports determining and implementing solutions to the threats. The security assurance process ensures that the security features (high-level security requirements that express protection capabilities of the software to mitigate the threats practices, procedures, and architecture of software accurately mediate and

enforce the security policy [7]. Security policies state the required protection of the information objects; they are rules for sharing, accessing, and using information, hardware and software [6].

2.3 Security Assurance Cases

Security assurance enables developing coherent objective arguments which could be reviewed to support claiming that a software product mitigates its security risks. A security assurance case [13], a semi-formal approach for security assurance, is a collection of security-related claims, arguments, and evidences where a claim, i.e. a security goal, is a high-level security requirement, an argument is a justification that a set of (objective) evidences justify that the related claim is satisfied, and an evidence is a result of a verification through, for example, security testing, source code security review, mathematical proofs, checking use of secure coding standards, qualification of the developers in terms of training on developing secure software etc.[6] The main steps of creating a security assurance case in sequence are:

Identify the claims—decompose the claim "the software is secure" into sub-claims such that satisfying the sub-claims induces satisfying the claim. The sub claims (which in turn become claims) could be iteratively subdivided, until getting verifiable sub-claims.

Establish the context—specify additional information for claims, such as definitions, reference to documents, explanations, and assumptions.

Identify the strategies—provide information on how a claim is decomposed into sub-claims. The strategy could be explicitly described in the assurance case or be implicit if no strategy is specified.

Identify evidences—collect the result of using the security evaluation techniques, such as security testing and security review of source code to evaluate the security countermeasures [13] used to eliminate or reduce the risk of the threats to the software and achieve the related security goals.

Specify the arguments—show implicitly or explicitly that evidence supports a claim. For example, the results of a security analysis tool may report that the software has a set of code security vulnerabilities (e.g., buffer overflow). The argument describes that the "errors are false positives and the claims satisfied. Now we are going to discuss new security requirements may demand a part release, new release, or to be an emergency case. The latter implicates that all ongoing development will be stopped until the emergency level is reduced.

2.4 The Risk Analysis Component

Security assumes that an attacker can outwit and bypass thoughtful protection and may anticipate vulnerabilities that have not yet occurred, i.e., outside the scope of the current risk analysis. Risk can be defined as the combination of the probability that a threat will occur and the possible consequences it may impair. In combination, this means that risk analyses should include not only known or established risks, but also newly discovered risks and risks that change over time, i.e., the probability and/or the consequences changes due to the protection included in the system or product under review. In the industrial setting of our study, the risk analyses are executed by the development team headed by the security master.

2.5 Risk Estimations

The probability of a successful attack has a range between 1 and 5, where 1 signifies a low likelihood of an attempt and success of an attack and 5 a high probability of an attack. Similarly, the impact of the attack also has a range between 1 and 5, where:

1. Minor damage to the system or service from internal (safe) source and data.
2. Minor damage to the system, service or data from external source, e.g. network connections.
3. Denial of service for other users or circumvention features.
4. Disclosure of confidential data, destruction of data, escalated privileges or financial fraud.
5. Full system (or component) compromised or undetected modification of data.

2.6 Actions Based on the Risk Analysis Results

A risk analysis is executed ones per BUC during the design phase of the software development process. Each risk analysis focuses on the requirements of that specific BUC only. Depending on each obtained risk value and the risk's relevance to the BUC, one of three actions is determined:

To carry out the correction within the BUC as an US that mitigates the risk.

To create a new BUC, delaying the correction for future versions

Accept the risk. The security master might aid the team with the task to manage it, but it is the team who owns the risk. A technical solution is proposed and implemented with the aim to mitigate the risk. Often, developers derive the security requirements from the functional requirements as implicit security requirements (14). Our hypothesis is that both the number of risks and the ability to address them will increase when secured enhanced agile software is used compared to the traditional process. Furthermore, we analyze whether the extra resources spent on security is worth the potential improvement.

3. Proposed Secured Agile Software Process Model

Agile software development transpires fast. The high frequency of iterations and releases often translates to wildly dynamic application build structures with new components or apparatus added regularly throughout the software development life cycle (SDLC). Risk assessments are often only implicitly part of functional discussions. While not necessarily as explicit security iteration as suggested in literature [10], customers and developers should explicitly address the risks and the non-functional security requirements early in the project to have a common development for improved security in projects of average security-criticality [2]. From the block diagram it can be observed that in the requirement phase we can enroll the specific analysis and review the security with proper guidelines according to the demands of customer's satisfaction. In the development to the engineer we must be aware of the tool design of software. Before designing a tool of software for a particular purposes, we must be confirmed the research and simulation in a virtual environment. We can add some features in the tool design sector to improve the security of the software by adding the application of specific structure, internal and external review, secured design principles, informal validation, formal validation, and cryptography and network security concepts for web based software process development. In the implementation and software development sectors we can add some other features to improve the security factors of software by adding requirement testing, security evaluation, and test depth analysis, integration testing, vulnerability testing, change authorization of that's particular conditions as well quality control and quality testing must be concerned. To implement the software in market we must be aware of the service, feedback of the software, maintenance and modification of the system to provide the proper delivery of software.

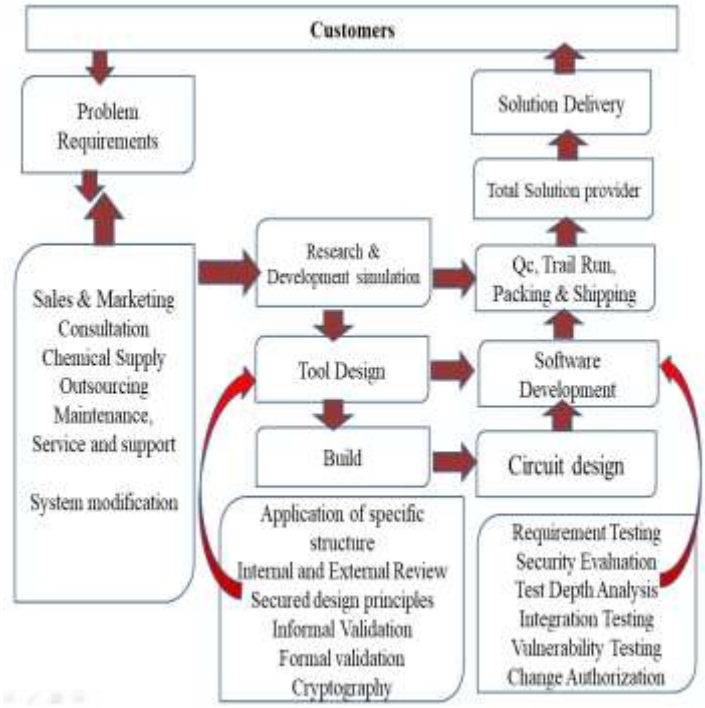

Fig.2: Block diagram of Proposed Secured Agile Software Process Model

4. Initial Evaluation

From the empirical analysis and block diagram we can see that here some additional features has been added to the proposed model for that we get some values that has been shown below:

	Model	
Attributes	**Agile**	**Improved Agile**
Accuracy	45	60
Automation	50	75
speed	60	78
Maintainability	65	80
Elasticity	50	65
Vulnerability	70	40
Change Authorization	60	35

Table 1: Agile and Improved Agile Model

From the table if we draw the graph then we can get the following graph that has been mentioned above:

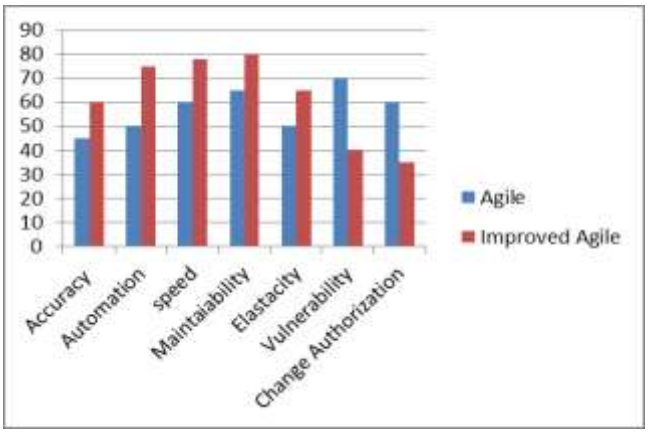

Fig 3: Performance analysis of Agile and Improved Agile Model

From the graphical analysis we can see the performance of different attributes of our proposed model compared with the existing agile model. It has been observed that our proposed model will improve the performance of agile software process model. Agile software process is a plan driven process where all the accomplishments are scheduled first and the improvement is measured according to the plan. Moreover agile process planning is incremental and it's easier to change the process to reflect requirement changes at each phases of software process. Agile method is an approach to the project controlling which supports to respond to the unpredictability of building software through incremental, iterative work cadences known as sprints. Agile model is one of the best methods to implement any changes requests because amount of uncertainly is very less complicated compared to other software process model. One of the differences between agile software development methods and waterfall is the approach to quality and testing. In the waterfall model, there is always a separate testing phase after a build phase; however, in agile software development testing is completed in the same iteration as well as security is also considered. We think that our proposed model will also improve the overall performance of the agile software process model since security concern is the major issue in our proposed system. Especially in the web based application, we can use different crypto system to secure the system more and more. Moreover if the visual security process can be developed, the agile software process model will be more benefitted.

5. Future Scope

In contrast to traditional enlargement of software process, agile development scope along with security is an extra lever that can be readily manipulated according to the customers' feedback continuously through graphically using simulation, process modeling, and system dynamics scope adjustment. This system of software process will also provide the continuous support of security related to the security concern of agile software model. Different types of attack of software process can be also minimized in a certain level so that actual output and expected output of the software process will be equivalent. Adjustment with other software process model is also considered according to the environment of setting issues. So We can say that our proposed software process model will increase the performance of overall software process model.

6. Conclusion

This paper concludes that the agile software development approach does not prevent ensuring the security of software increments produced at the end of each- iteration. It proposes a method for security assurance of software increments and integrates security engineering activities into the agile software development process. The method enables ensuring the delivery of secure software at the end of each-iteration. The main advantages of the approach are: helping reduce the cost of reassurance of software security, and helping reduce the cost of mitigating threats. The reason for the first advantage is: the development team could reuse parts of the security assurance cases in the assessment of the new increments. The reason for the second advantage is: the approach ensures that security goals achieved in the early iterations of the development are preserved in the subsequent software increments. The main limitations of the current method are: It is not scalable enough; it requires extra cost and it applies to modular software, where security claims is associated with specific component as well as It is applied using graphically/visually. This paper demonstrates the possibility to develop secure software using the agile development process through a simple case study the methods we propose. Our future work include automating the proposed development process and security reassurance method, evaluating the cost of security reassurance on the

development time, and investigating how to consider security design principles such as the fail safe principal in the assurance.

7. Acknowledgement: Foremost, I would like to express my sincere gratitude to my supervisor Ziaur Rahman, (Assistant Professor, Department of Information and Communication Technology, Mawlana Bhashani Science and Technology University, Santosh, Tangail-1902, Bangladesh) for his continuous support, motivation, enthusiasm, and immense knowledge. His guidance helped me in all the time of research and writing of this thesis.

8. References

[1] Dejan Baca1, Martin Boldt2, Bengt Carlsson2, and Andreas Jacobsson "A Novel Security-Enhanced Agile Software Development Process Applied in an Industrial Setting" IEEE ARES International Conference,2015

[2] Siponen, M., Baskerville, R., and Kuivalainen, T., "Integrating Security into Agile Development Methods", Proc. of the 38th Hawaii Int. Conf. on System Science, 2005

[3] Konstantin Beznosov, Philippe Kruchten," Towards Agile Security Assurance" University of British Columbia

[12] K.M. Goertzel and T. Winograd, "Enhancing the Development Life Cycle to Produce Secure Software,"Technical Report DAN 358844 Defense Technical Information Center (DTIC), https:// buildsecurityin.us-cert.gov/bsi/resources/1185-BSI/1191-BSI. html,, 2008.

[13] J. Vivas, I. Agudo, and J. Lpez, "A Methodology for Security Assurance-Driven System Development," Requirements Eng., vol. 16, no. 1, pp. 55-73, 2011.

[14] H. Chivers, R. F. Paige, and X. Ge. Agile security usingan incremental security architecture. In H. Baumeister, M. Marchesi, and M. Holcombe, editors, XP, volume 3556 of Lecture Notes in Computer Science, pages 57–65. Springer,2005.

[15] K. Beznosov and P. Kruchten, "Towards Agile Security Assurance," Proc. Workshop New Security Paradigms (NSPW '04), pp. 47-54, Sept. 2004.

[16] J. Wayrynen, M. Boden, and G. Bostrom, "Security Engineering and eXtreme Programming: An Impossible Marriage?" Proc. Fourth Conf. Extreme Programming and Agile Methods, pp. 117-128, Aug. 2004.

[17] K.M. Goertzel and T. Winograd, "Enhancing the Development Life Cycle to Produce Secure Software,"Technical Report DAN 358844 Defense Technical Information Center (DTIC), https:// buildsecurityin.us-cert.gov/bsi/resources/1185-BSI/1191-BSI.html,,2008.

[18] M. Boberski, J. Williams, and D. Wichers, OWASP Application Security Verification Standard 2009. The Open Web Application Security Project (OWASP), https://www.owasp.org/ OWASP_ASVS_2009_Web_App_Std_Release.pdf, June 2009.

[19]http://www.ambysoft.com/essays/agileLifecycle.html

[4] Kjetil Moløkken-Østvold and Magne Jørgensen," IEEE Transactions On Software Engineering" Vol. 31, No. 9, September 2005

[5] Mikio Aoyama," Web-Based Agile Software Development" IEEE Software November/ December 1998

[6] H. Chivers, R. F. Paige, and X. Ge. Agile security using incremental security architecture. In H. Baumeister, M. Marchesi, and M. Holcombe, editors, XP, volume 3556 of Lecture Notes in Computer Science, pages 57–65. Springer, 2005

[7] 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.

[8]https://en.wikipedia.org/wiki/Agile_software_development [last access , 10 feburary, 2018]

[9] R. Shirey. Internet Security Glossary, Version 2. RFC 4949 (Informational), http://www.ietf.org/rfc/rfc4949.txt, Aug. 2007

[10] J. Vivas, I. Agudo, and J. Lpez, "A Methodology for Security Assurance-Driven System Development," Requirements Eng., vol. 16, no. 1, pp. 55-73, 2011

[11] K. Beznosov and P. Kruchten, "Towards Agile Security Assurance," Proc. Workshop New Security Paradigms (NSPW '04), pp. 47-54, Sept. 2004.