ISSN:0975-9646

Analysing Security and Software Requirements using Multi-Layered Iterative Model

Sonia[1], Archana Singhal[2], Jyoti Balwani[3]

[1,3] *Department of Computer Science, University of Delhi, Delhi, India*
[2]*Department of Computer Science, University of Delhi, IP College for Women, Delhi, India*

***Abstract*— Nowadays, security is of great concern for any organization developing software systems for various requirements. Moreover, the same becomes more complicated during integration of security measures with agile software development methodology due to its lightweight informal nature. The requirements engineering is considered as one of the key element associated with any software development process. This motivates us to suggest a FLAMIRA model that provides seamless integration of security needs with software requirements in an iterative manner. In agile processes, requirements are recorded in the form of user stories developed jointly by customer's representative and the development team. User stories are useful for agile processes as they define requirements using a low-cost, user centric and flexible approach. Keeping this aspect in mind we are integrating abuser stories for security requirements with user stories. FLAMIRA is a multi-layered model which shows us the path to be followed right from the identification of the user stories till the formulation of abuser stories. This paper concludes with a set of user stories and abuser stories to be followed in each iteration.**

***Keywords*— Agile Methods, User Stories, Abuser Stories, Requirement engineering.**

I. INTRODUCTION

In the modern business environment, it is difficult to keep pace with ever changing requirements of software development. Thus, agile processes have emerged as most promising software development methodology which is adaptive in nature and is able to meet the challenges posed by traditional development methods. According to some research papers [1,2] implementation of security is not very much effective in agile processes since security activities are lengthy and require too much documentation. In contrast, agile principles define iterative and short development life cycles having main emphasis on direct communication between its customer and the developer. The main principles which are followed by agile processes are defined by agile alliance [4] and manifesto for agile software development [5]. The user story format is advocated in agile methods due to its lightweight, informal nature which can be easily specified by customers in their natural language. Thus to specify software requirements we suggest user stories.

Security in software development must be given highest priority to counter the threats posed by attackers. It can be achieved by mapping security techniques with agile development right from the beginning of software development. Thus, in this paper, we have proposed a multi-layered model, FLAMIRA (Four Layered Agile Model for Iterative Requirements Analysis) which works on the requirement engineering phase of software development lifecycle. The first layer recorded software requirements as a set of user stories for the complete project. Then successive layers suggested how these user stories can be managed in an incremental and iterative agile software development. Last layer incorporates security requirements in the form of abuser stories. In this model, the iterative nature of agile software development has not been overlooked even during security requirements specification. The key feature of the proposed model is that it facilitates iteration planning and supports a shared integration of security and software requirements within each iteration.

The rest of the paper is organized as follows. Section II gives brief description of the related work. Section III summarizes the concept of user stories and abuser stories in security requirements engineering and brief overview of agile software development. Section IV presents our proposed model FLAMIRA for iterative software and security requirements analysis. A case study and implementation of proposed model is described in Section V. The overall conclusion and directions for future work is given in Section VI.

II. RELATED WORK

The concept of security requirements engineering is widely available in literature. It describes that for integrating security requirements with software development life cycle, the requirement engineers must discover security requirements along with software requirements. Many researchers have contributed in various ways to integrate security and agile processes [6, 7, 8, 9]. Danier Mellado has given a comparative study of proposals for establishing security requirements for development of secure software system [10]. Howard Chivers had delivered a quality work on agile security using incremental security architecture and also showed agile development for secure web applications by integrating risk assessment with agile processes [11, 12]. John Peeters has put forward the idea of using abuser stories explaining how attacker may abuse the system and jeopardize stakeholder's assets with usual user stories ranked according to perceived threats [10, 13]. Vidar Kongsli has also given security in web applications using misuse stories with user stories to capture malicious use of attacks [8].

They all suggested various techniques to implement security in agile processes. Need for further refinement in

this area motivates us to develop FLAMIRA model. Many methods are in use to specify security requirements but none of the above presents a roadmap which can guide developer about the step by step procedure to be followed going from software requirements to security requirements. Focusing on iterative nature of agile software development our model also categorizes user stories and abuser stories in different groups which help us to know which security practice is to be implemented in which iteration.

III. BACKGROUND

This section presents in brief the concepts used in the proposed model.

A. Agile Requirements Analysis and Planning

During software development requirements emerge and evolve as software is developed. Agile requirements are developed in small, bite-sized pieces. For agile methods user stories are appropriate for describing features and functional requirements of the software to be built. A user story is a high-level description of the requirements that will be valuable for the user or owner of the software.

Abuser stories describe the undesired behaviour of the system and for analysing security requirements they play a vital role. Thus, for secure software development, abuser stories are collected by developer from stakeholders on index cards. These abuser stories are based on given user stories, assets and security objective of the given system. After that remaining abuser stories must be created by developers and security experts based on their past experience.

B. Agile Software development

Agile software development (ASD) facilitates software development at fast pace and ever-changing. Agile processes promote iterative and incremental development, minimum documentation and customer satisfaction. In comparison to traditional methods, ASD is adaptive in nature supporting continuous changes t any stage of software development. ASD consisted of various methods. Some of them are Extreme Programming (XP), Scrum, Feature Driven Development (FDD) and so on. In ASD requirements are implemented on priority basis due to its short development lifecycles. This feature provides iterative nature to ASD. Thus in ASD most important requirements are confirmed by user at the starting of each iteration and those requirements are arranged systematically for implementation in next iterations. In ASD active user involvement and cooperation, collaboration, and communication between all team members is essential.

IV. PROPOSED MODEL FLAMIRA FOR ITERATIVE SOFTWARE AND SECURITY REQUIREMENTS ANALYSIS

Requirement phase is the most important or crucial phase of software development. If the interpretation of requirements in earlier stages of software development is not systematic then later stages may also suffer. As we are considering software development through agile methodology, the issues related to short release cycles or iterations can't be neglected during management of software and security requirements management. Therefore, in this section, we propose a Four Layered Agile Model for Iterative Requirements Analysis (FLAMIRA), as shown in Fig.1. This FLAMIRA provides step by step iterative procedure to identify agile software and security requirements.

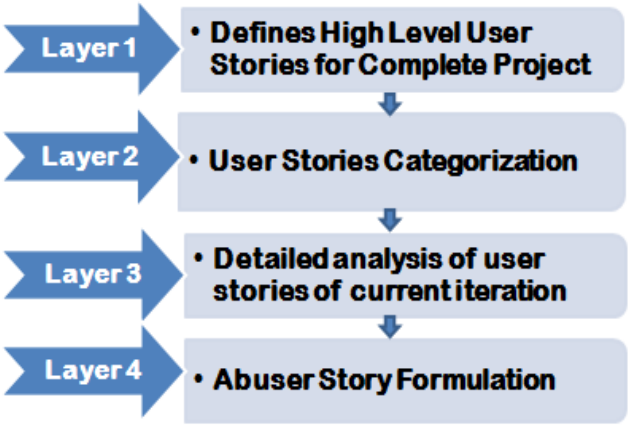

Fig.1. Four Layered Agile Model for Iterative Requirements Analysis (FLAMIRA)

A. Layer 1: Defines High Level User Stories for Complete Project

This layer identifies high level user stories that define the scope of whole project under development. As agile software development is iterative in nature, the proposed model, at this layer, includes requirements of complete system in first iteration. In successive iterations it includes requirements which are yet to be analysed for the whole system. Activities performed at this layer are

1) Initially, requirements in the form of user stories (what the user wants to achieve from that project) are captured from the end user for the complete project. These user stories discussed above are written on an index card which contains small sentences in natural language and describes the intent of the story. The most common format of user story is

" As a <Role> i want to <Action> so that <Result>"

2) Then, face to face communication between development team, customer, end users and other stakeholders clarifies the details of user stories and estimates the business value of each user story (rank a user story according to perceived value by the customer) from the set of user stories for the complete software. Business values of these stories can vary in each iteration. Like in one iteration, customer considers a particular user story important but that user story may become less or more important in next iteration on the basis of project growth.

3) This layer reconsiders requirements for the remaining part of the software development process after each iteration. At the start of each iteration, this facilitates the customer to add new stories, change the value of existing story, split stories or eliminate a story [14].

B. Layer 2: User Stories Categorization

After defining user stories the development team analyses each user story at this layer and assigns

- a cost, measured in development weeks, to each story.
- an iteration number, a number that describes in which iteration the user story is to be implemented in software under development.
- estimated delivery date to each user story.

Before assigning iteration number, the team will decide number of user stories to be implemented in each iteration. However, this number is not fixed. If stories to be implemented are lengthy they will consume more time and less number of stories will be implemented in that iteration. Keeping the complete project in mind tentative iteration numbers are assigned based on the output of first layer. However, this number will be revised at the beginning of each iteration for the remaining project.

After getting business value from layer 1 and tentative iteration number from layer 2, these user stories are categorized into different groups.

In the proposed model, we eliminated the need of onsite customer or end user participation at this layer. This is the advantage of FLAMIRA model since we have considered only those activities at this layer which don't require customer collaboration. The motivation behind this is to reduce customer presence which saves customer's time. We have sometimes felt that customers don't want to go for agile software development due to all-time presence constraint. Our aim is to involve customer only whenever necessary which relaxes the customer from the software development burden.

C. Layer 3: Detailed analysis of user stories of current iteration

The user stories to be implemented in each iteration have been decided at layer 2 by the developer. Here, user stories are discussed by the developer in collaboration with customer to clarify and identify the detailed requirements for the next iteration. Ultimately, agreeable user stories are collaboratively established by customer and product developer for the next iterations [15].

At this layer, only user stories to be implemented in current iteration have been analysed by the developer team and the customer. Analysis take care some points like

1) There must be minimum interdependency between stories as far as possible. Since, this dependency makes the development process complex and also increases its development time.
2) The user stories must be flexible. Index cards of user stories describe short description of functionality. However, detailed planning of its implementation part should be left on the developer.
3) The proper estimation of the size of each user story under consideration. Like, if cost of implementing user story is exceeding the decided value than that user story card is returned to the customer. The customer with the development team as per their requirement will split that user story.
4) After analysing user stories describing features and functional requirements of the software to be built, this layer will identify critical assets of the system keeping its security in mind. These assets will be helpful in describing the threats in next layer.

After analysis, if customer finds some features that must not be implemented in current iteration or vice versa then reshuffling of stories can be done at this layer.

At the end of this layer, an acceptance criteria or test must be written by the customer with each user story before the same is implemented at this layer. This is necessary for testing the goals specified by user story, whether fulfilled or not.

D. Layer 4: Abuser Story Formulation

To develop a secure software development process, security requirements are essential to be identified in proposed FLAMIRA model. The assets identified in layer 3 of FLAMIRA will serve as an input to determine security requirements in the form of abuser stories. The abuser stories describe the undesired behaviour of the system in contrast to user stories which describe the desired output from the system. The stepwise iterative process of identifying and integrating abuser stories with user stories is described below

1) Initially, at this layer, the developer writes some abuser stories on index cards using assets and security objectives (like goals, constraints with user stories) in collaboration with customer and stakeholders. The abuser stories are written in the same way as user stories. These abuser stories are based on user stories as well as assets of given system and can be seen as agile counterparts of abuse cases or misuse cases [3].
2) Some stories are not related to user stories identified above but they show attacker's potential intentions of damaging the assets. These stories are also placed in category of abuser stories and identified in this step.
3) In case of abuser stories, ranking or scoring has been done on the basis of perceived threats posed by them to customer's assets. The ranking is based on severity, risk factor, impact and need of the user stories. Abuser stories also carry a cost similar to user stories which amounts to negative business value [13].
4) As in case of user stories, abuser stories are also implemented iteratively in parallel to user stories in proposed FLAMIRA model. For iterative development of software, abuser stories must be categorized into several iterations. But before categorizing, we must consider abuser stories for the whole system, as was done for user stories in layer 1. Then, based on rank and cost assigned in step 3 for abuser stories, grouping of these stories into different categories is done and iteration number to each story is assigned.
5) Now, the developer knows that in which iteration a particular user story is to be implemented. Thus, instead of focusing on overall system, developer just

plans for current iteration. It helps him to achieve a particular release on time securing well defined features. Baskerville et al. has also said that when the development is carried out in several development releases, the developers should be informed in which release the abuse case is prevented (i.e. countermeasure is implemented) [12]. Abuser stories are analysed in the same way as we have analysed user stories in layer 3. Moreover abuser stories are also revised at the starting of each iteration during the planning phase and may be updated if necessary. Any number of abuser stories can be added, deleted, split or combined at the start of each iteration.

V. CASE STUDY AND ITS IMPLEMENTATION

In this section we go through the development stages of the suggested FLAMIRA model using a case study "Automated Teller Machine" (ATM). It is a network technology that provides several banking services to customer and security, therefore, becomes a major concern.

We have developed an automated Requirements Analysis tool (RAT) for implementing the layers of our FLAMIRA model. To know the efficiency of proposed model, we have implemented RAT on the above said case study. Suggested tool has been designed in Java NetBeans IDE 6.5.1. MYSQL has been used to store the data of our case study. Designed screens are user friendly and self explanatory.

According to our proposed model, the first step to build an ATM security system is to identify user stories for the complete project. Some of the user stories for the ATM system are

- ***As an* ATM user *I want to* withdraw money from my bank account *so I can* increase my cash on hand.**
- ***As an* ATM user *I want to* check balance in my bank account *so that I can* withdraw or deposit sufficient money.**
- ***As an* ATM operator *I want to* restock the ATM with money *so the* ATM will be available for customers to withdraw funds.**
- ***As a* Bank Business Owner *I want to* set the ATM's withdrawal parameters *so the* ATM will provide funds to customers but protect against fraudulent activities.**

Fig. 2 Main Window

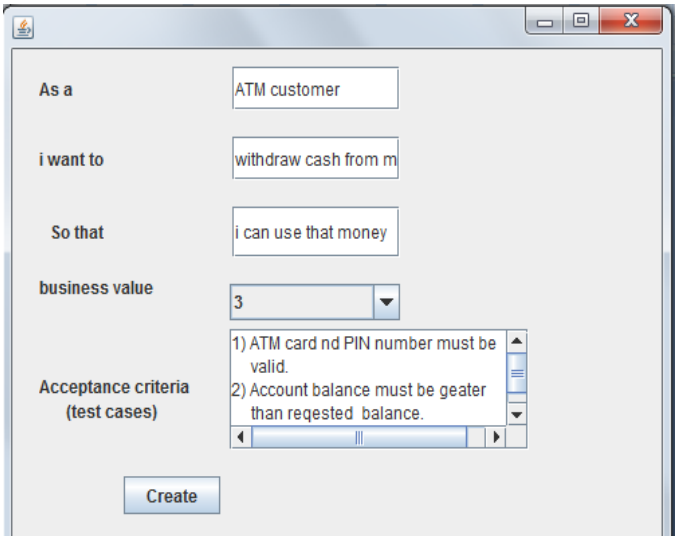

Fig 3 Template for storing User and Abuser Stories

To store these user stories in the tool, the developer will choose the 'User Stories' option from the main window as shown in snapshot of Fig.2. Then, the developer or customer provides details for each user story in the relevant fields as shown in Fig. 3. Any number of user stories can be stored in the RAT using the same procedure.

Then according to layer 2, developer will complete the cost, assets and tentative iteration number of each user story. To provide these values, developer will first choose assets option from view button of main menu. Then required values can be given in the template as shown in Fig. 4. Next, the developer can analyse all the user stories at a time using the button 'show stories and related information' as given in Fig. 4. After analysis, the developer can take appropriate actions like changing iteration number, splitting of stories as suggested in layer 3 of proposed model.

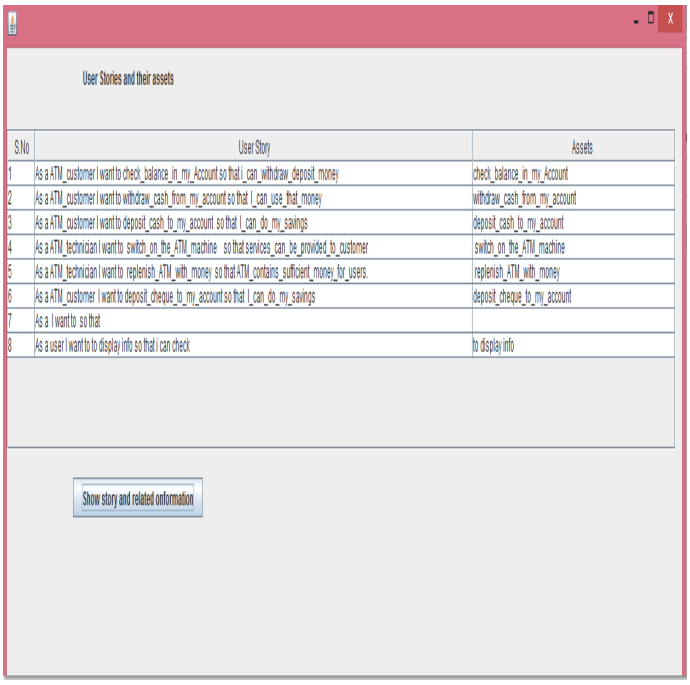

Fig. 4 Template for storing Assets and showing all User Stories

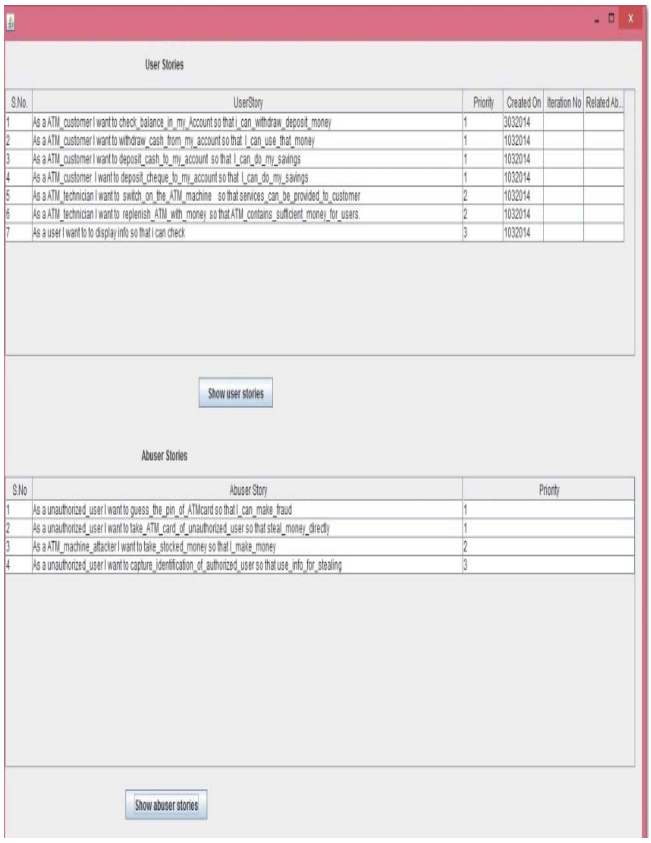

Fig. 5 Summarized View of all User and Abuser Stories with Details

We can store abuser stories by choosing 'Abuser Stories' option from main window. For our case study, we have stored abuser stories of ATM. Here our main assets are money and user's secret information like pin number. Some of the abuser stories for ATM are

- ***An* unauthorized user *wants to* capture identification of an authorized user *so that* he can use that information for stealing money.**
- ***An* unauthorized user *wants to* take ATM card of an authorized user without the authorized user knowledge *so that* he can make copy of it or use the card directly for stealing money.**

To get summarized view of all the user and abuser stories with their complete details, the developer will select 'Created Stories' option from the view button of main menu as shown in Fig. 5. This view is useful to analyse all these stories simultaneously with detailed information's like

- **The deadlines for completion of user stories.**
- **Which abuser story has been derived from the corresponding user story?**
- **In which iteration a user story or an abuser story is to be implemented?**

VI. Conclusion and Future Work

Agile methods are extremely popular in software development companies as these methods are informal and lightweight in nature having short time spans. However, integration of security in agile methods is still a challenging task for the organizations. Thus, in this paper, we have proposed a multi-layered iterative model FLAMIRA which can analyse security requirements along with software requirements. The proposed model has used user stories to identify the software requirements and abuser stories to get security requirements. This model is able to achieve better collaboration, communication, customer satisfaction and concise light weight documentation for better integration and iterative development. For future work of any system software, one can consider more advance techniques to integrate security in agile methods. Also, scope of the present tool can be expanded to recommend more security activities in various agile methods.

References

[1] Beznosov, K. Extreme Security Engineering: On Employing XP Practices to Achieve 'Good Enough Security' without Defining It. First ACM Workshop on Business Driven Security Engineering (BizSec), Fairfax,VA, 31 October, 2003

[2] Wäyrynen, J., Bodén, M. & Boström, G. Security Engineering and eXtreme Programming: An Impossible Marriage? In Proceedings of the 4th Conference on Extreme Programming & Agile Methods. 2004, Springer-Verlag, Lecture Notes in Computer Science. p. 117 .

[3] Sonia, A.Singhal, "Development of Agile Security Framework using a Hybrid Technique for Requirements Elicitation", International Conference on Advances in Computing, Communication and Control (ICAC3) 2011, Mumbai, India, Volume 125, Part1, pp. 178-188.

[4] The Agile Alliance Home Page, http://www.agilealliance.org/home.

[5] Beck, K., et al. (2001).Manifesto for Agile Software Development, February 2001.

[6] Siponen, M., Baskerville, R., Kuivalainen, T.: Integrating security into agile development methods. In: 38th Annual Hawaii International Conference on System Sciences, 2005.

[7] Daud, M.I.: Secure Software Development Model: A Guide for Secure Software Life Cycle. In International MultiConference of Engineers & Computer Scientists, Hong Kong, 2010.

[8] Kongsli, V.: Towards Agile Security in Web Applications. In the Proceedings of OOPSLA October 22-26, 2006, Portland, Oregon, USA, ACM.

[9] Boström, G., Wäyrynen, J., Bodén, M., Beznosov, K.: Extending XP Practices to Support Security Requirements Engineering. SESS'06, May 20–21,2006, Shanghai, China.

[10] Mellado, D., Fernández-Medina, E., & Piattini, M. A Comparative Study of Proposals for Establishing Security Requirements for the Development of Secure Information Systems.M. Gavrilova et al.: ICCSA 2006, LNCS 3982, 2006. Springer-Verlag Berlin Heidelberg 2006.

[11] Ge, X., Paige, R.F., Polack, F., Chivers, H., Brooke, P.J. : Agile Development of Secure Web Applications. ICWE'06, July 11-14, ACM.

[12] Baskerville, R., Levine, L., Pries-Heje, J., Ramesh, B. and Slaughter, S. Is Internet speed Software Development Different? IEEE Software (20:6), 2003, pp. 102–107.

[13] Peeters, J.: Agile Security Requirements Engineering. In: Requirements Engineering for Information Security, 2005.

[14] Pressman, R.S. (2005). Software Engineering A Practitioner's Approach, McGraw Hill, 2005.

[15] Paul E.McMohan. 2005. Extending Agile Methods: A Distributed Project and Organizational Improvement Perspective