Selection of Security Activities for Integration with Agile Methods after Combining their Agility and Effectiveness

Sonia, Archana Singhal
University of Delhi
Delhi, India
{soniacsit, singhal_archana}@yahoo.com

ABSTRACT: *Nowadays security concerns are central in software engineering. Moreover, the idea of incorporating security in agile software development has always been difficult because most of the security activities conflict with the agile principles. Agile development methodologies promise simple and iterative development with minimal documentation and a solution to rapidly changing environment. However, security techniques are complex and require too much documentation thus not suited for agile methods. The goal of this paper is to select those security activities which are best to integrate with agile methods. This selection is based on two distinct measures, effectiveness and agility degree of security activities. The former indicates that the considered security activity will produce the intended result and ultimately reflects its impression in achieving security of software system. The latter refers to the flexible and low-cost ability of the security activity in providing effective responses to unpredictable changes. The challenge is to find the right mix of above said two measures according to project's need. This has been accomplished in the proposed approach, CEASA (Combining Effectiveness and Agility of Security Activity), by using a weighted decision theory.*

Keywords: Security Activities, Agility Degree, Agile Software Development Methodologies, Weighted Decision Theory

Received: 11 January 2014, Revised 14 February 2014, Accepted 24 February 2014

1. Introduction

In recent years, researchers have started implementation of security techniques right from the beginning of software development to get secured system software. These techniques have spurred their usage in agile methods but security activities always have conflict with the agile principles. Agile methods promise iterative, incremental and minimum documentation to deliver working software in short duration. But as indicated in some research papers [4, 5], implementation of security is not as effective in agile methods as it should be. Thus, security activities that are appropriate for integration with agile development methods need to be explored from the set of activities defined in secure software development processes.

This has motivated us to propose a new approach, *CEASA*, which suggests various security activities for integration with agile methods based on their agility degree and effectiveness.

Agility in general, is the ability to provide effective response to expected or unexpected changes. It can deliver working software in short duration. Effectiveness of security activities can be defined as a measure that will fulfill the requirement of efficient risk removal. In this approach, *CLASP* security-related activities are considered for integration. *CLASP* is a secure software

development process that contains best practices and security activities for providing security with the software development life cycles in a structured way [6, 10].

The said approach starts with the computation of agility degree of the considered security activities. For this agility degree calculation, a *FISA-XP* approach suggested in our previous work [1] has been used. This *FISA-XP* approach provides some resultant security activities that could be the best for integration with agile methods keeping agility of developing software as high as possible. Agility degree measurement is the first step of our approach that can be done using *FISA-XP* approach. Then the resultant security activities that we get from first step are further considered in our second step for measuring the effectiveness of security activities. The effectiveness can be measured either by qualitative approach using a *MOD* analysis [2] or by quantitative approach using a hypothesis testing technique [3].

After computing the agility degree and effectiveness, next step is the development of *EASAST* table (Effective Agile Security Activities Selection table). This table records the value of agility degree and effectiveness. Thereafter, using security activity lookup table (*SALT*), the Combined Effective Agile Value (*CEAV*) has to be determined. Then, the *EASAST* table is sorted on the basis of *CEAV* function. After sorting these activities in the table a cutoff line is assigned which means that only those security activities which lie above this line will be considered for integration and the rest have to be discarded. To determine the combined impact of effectiveness and agility of security activity during integration (or we can say the position of cutoff line), we have used a weighted decision approach. Here, decision is done by client by selecting the appropriate options from the list according to project needs.

The rest of the paper is organized as follows. Section II describes the related work that already exists in this field. Section III presents a brief summary of agile software development and the *CLASP* (Comprehensive Lightweight Application Security Process) security development process. Section IV describes our approach by providing the steps to be followed for integrating security activities with agile processes. Finally, Section V presents overall conclusions and some guidelines for future work.

2. Related work

The secure software development literature has become very vast over the past decade. Lot of research has been done which suggests many techniques for implementing security activities with agile methods [7, 8, 9]. Dejan Baca considered the security activities of three well-known security-engineering processes: Microsoft *SDL*, Cigatel Touchpoints and Common Criteria and compared these activities to identify their integration possibilities with agile development processes [11]. *CLASP*, *SDL* and Touchpoints have also been compared by B. D. Win [12]. In a similar manner, '*Beznosov has classified security assurance methods and techniques with regard to their clash with agile methodologies*' [13]. An iterative framework named Agile Security Framework (ASF) was suggested by us [14, 15]. It provides step-by-step guidance for applying security techniques wherein agility is maintained at each stage.

Several researchers have put forward the idea of agility measurement. Yang and Li had proposed a weighted index of agility where the agility intensity was weighed by its importance [16]. Agility Measurement based on Analytical Hierarchical Process (AHP) was suggested by Ren [17]. Yusuf et al. considered ten decision domains and derived related agility attributes using these domains [18]. In their approach, each of the hierarchy levels was evaluated by a paired comparison ranking. The comparison data was then converted to relative weights and the final score of the hierarchy could be calculated. A. Qumer has evaluated the degree of agility for six agile methods [19] and Sherehiy et al. presented a review of enterprise agility [20]. Walid Al-Ahmad has suggested method for integration of *CLASP* security activities in all phases of XP in [21]. An assessment of the relative contribution of facets of agility to describe systems development success using an AHP approach has been given by Saonee Sarker [22]. We have computed the relative contribution of agility features in the context of agile and security activities by proposing the WRIAF (Weights describing relative importance of agility features) approach [23]. Hossein Keramati integrated some security activities with agile methodologies from the agility point of view [24]. We have also proposed a *FISA-XP*, which can be adopted for the development of a secure software system [1, 25]. This approach integrates security activities with the core activities of Extreme Programming based on their degree of agility.

Some authors have suggested evaluation methods and measurement framework guides for data gathering from various sources like security organizations, technical personnel and other statistics available regarding security of the systems. [26,27,28]. Moreover, SEI (Software Engineering Institute) has suggested an analysis, called as *MOD* analysis that provides qualitative measures to decision makers for analyzing software security [29]. In this analysis, Christopher Alberts et al.assist developers by

providing measurement at each phase of the life cycle to assess whether required level of security has been achieved or not. Also, we have presented an approach to measure effectiveness of a security activity qualitatively and quantitatively. The effectiveness of security activities is measured by analyzing them qualitatively using MOD analysis [2] and quantitatively in terms of its risk removal efficiency using hypothesis testing approach [3].

Ayalew et al. in [30] have integrated security activities with agile methodologies after evaluating security activities in terms of cost and benefit. Literature review clearly shows that different authors and researchers have worked on different aspects of integration of security activities with agile activities but no one has integrated these two activities based on the agility degree and the effectiveness measures of security activities. In the proposed approach, we have measured agility degree and effectiveness of security activities separately and then combine these two measures using a weighted decision theory where decision is taken by a client according to the project's needs.

3. Background

This section describes certain useful concepts required to understand the presented approach.

3.1 Agile Development Process

Agile software development has been preferred by software developers during the last decade. The problem of dealing with heavily regulated, rigid traditional methodologies has been solved by agile methods. All agile methods are based on some general principles defined by agile alliance [31] and suggested in the manifesto of agile software development [32]. These methods are lightweight and adaptive in nature. Due to their flexible approach, they support changes at any stage during software development lifecycle. Some of the most common agile methods are extreme programming, scrum, crystal clear and adaptive software development.

Agile software development methodology, as the name specifies, is based on the concept of agility. In general, agility is the ability to provide effective response to change. It facilitates communication among team members and customers and aims to deliver working software in short duration. Various definition has been given for agility, among them, Qumer and Henderson Sellers [19] defines agility as ''*Agility is a persistent behavior or ability of a sensitive entity that exhibits flexibility to accommodate expected or unexpected changes rapidly, follows the shortest time span, uses economical & quality instruments in a dynamic environment and applies updated prior knowledge & experience to learn from the internal and external environment*''. In simpler terms Tisni J. Kurin in [33] defines agility as "*the ability of the process to successfully cope with changes in requirement*".

3.2 Security Activities & CLASP Security Process

The problem of incorporating security into software development process is getting worse with time. During software development process, security must be in everyone's mind throughout its development lifecycle. In modern world, all development methodologies recognize the importance of software security. Also, many well-known processes exist for development of secure software, the most common among them are '*Microsoft's SDLC*' (Security Development Life Cycle), '*OWASP*'s *CLASP*' (Comprehensive Lightweight Application Security Process) and '*McGraw's Touchpoints*'. All of these security processes provide a specific set of security activities that can be integrated in different phases of the security-development lifecycle. These activities also improve quality and performance of secure software system. As features possessed by *CLASP* are favourable for agile methods, this paper considers *CLASP* activities for integration with agile processes.

Basically, *CLASP* stands for Comprehensive, Lightweight Application Security Process. This process is to be followed for considering security practices right from the beginning of the project during any software development. As described in [6], "*CLASP is an Activity driven, role-based set of process components whose core contains formalized best practices for building security into your existing or new-start software development life cycles in a structured, repeatable and measurable way*". During software development project, *CLASP* Best Practices are the basis of all security-related software development activities. It has 7 best practices and 24 security-related activities.

4. Proposed Approach CEASA: Selects Security Activities for Secure Software Development on the basis of their Agility and Effectiveness

System security is a wide issue that includes many security activities to be taken care of during entire software development

lifecycle. Moreover, before integrating these security activities with agile processes, the effectiveness and agility degree of security activities are considered as two important measures. Future trends are towards secure software development that needs both effective and agile security activities. Thus, in this section, an approach *CEASA* (Combining Effectiveness and Agility of Security Activity) is proposed. In this approach, proper balancing between these two measures has been done using a weighted decision theory. Here, decision is taken according to project's needs of an organization. Proposed approach proceeds in several steps as explained below and shown in Figure 1.

4.1 Step 1: Computing Agility Degree of Security Activities

The incorporation of security into the *SDLC* begins with the computation of the agility degree of security activities since to combine the effectiveness and agility degree of security activities first there arise a need to determine their individual values.

Therefore, a *FISA-XP* approach is used for calculating the agility degree of security activities [1] in step 1. This approach first identifies several features that affect the agility of an activity. But all identified agility features can't be equally important in contributing towards the agility of an activity. Hence, in order to find the relative importance of each feature, an approach *WRIAF* (Weights describing Relative Importance of Agility Features) is used [23]. It provides weightage to each feature based on its agility contribution to the activity. After finding these contributing factors and their relative weights, the next step is to assign scores to the security activities based on their agility degrees. The scores are called '*Preferred agile values*'. The original formulation for computing the agility degree uses a scale of 0 to 4, in which higher values indicate more compatibility of the integrating activity with the feature and lower values represent a less compatibility between them.

Once the '*Relative weights*' and '*Preferred agile value*' for all agility features have been assigned, the '*intermediate agility degree*' has been computed using the given formula

where,
$$IAD_i = \sum_{j=1}^{n} (W_{ij} * A_{ij}) \quad (1)$$

IAD_i is the '*Intermediate Agility Degree*' of the security activity *i*, A_{ij} is the '*Preferred agile value*' of an attribute j of the security activity *i*, W_{ij} is the weightage factor of attribute *j* of security activity *i*, that is, the '*Global relative weight*' of an attribute as given in column 5 of Table 1.

After calculating *IAD*, certain adjustment factors for finding the agility degree of the security activity SA_d has been used in the given approach. These adjustment factors also affect the agility degree of security activities to a certain extent. In this, the first factor is the '*involvement of a security expert*'. If an expert is fully involved then definitely the security implementation would be much faster and more accurate. Second factor considered here is whether a security activity is used frequently or not and how familiar the developers are with that activity. Since if developers have a clear understanding of implementing an activity then they can work on it more easily, quickly and correctly as compared to others.

The grades assigned are summed up giving a total of *M*, which ranges from 1 to 5. Thus, the agility degree of the ith security activity SA_{di} is computed by using the formula given in equation 2.

$$(SA_d)_i = \sum_{j=1}^{n} (W_{ij} * A_{ij}) * 0.1\ M \quad (2)$$

In the proposed approach, five *CLASP* security activities are considered for integration with '*Planning activity*' of Extreme Programming (*XP*) method. *XP* is one of the most popular agile methods.

After computing the agility degree of security activities, developers use an *AARF* factor. *AARF* can decide how much reduction in agility degree is acceptable when that security activity is combined with the agile activity. Resultant security activities are considered further for our proposed approach.

4.2 Step 2: Computing Effectiveness of Security Activities

The next step of the proposed approach is to measure effectiveness of security activities for secure software development process qualitatively and quantitatively. Qualitative effectiveness of security activity has been measured using *MOD* analysis [2]. It describes a way for conducting systematic analysis of interactively complex software reliance system. This approach first identifies the set of factors called drivers used to measure effectiveness in relation to its mission and objectives. Then, security activities are considered as drivers and an associated question related to each driver has been framed. To answer each driver

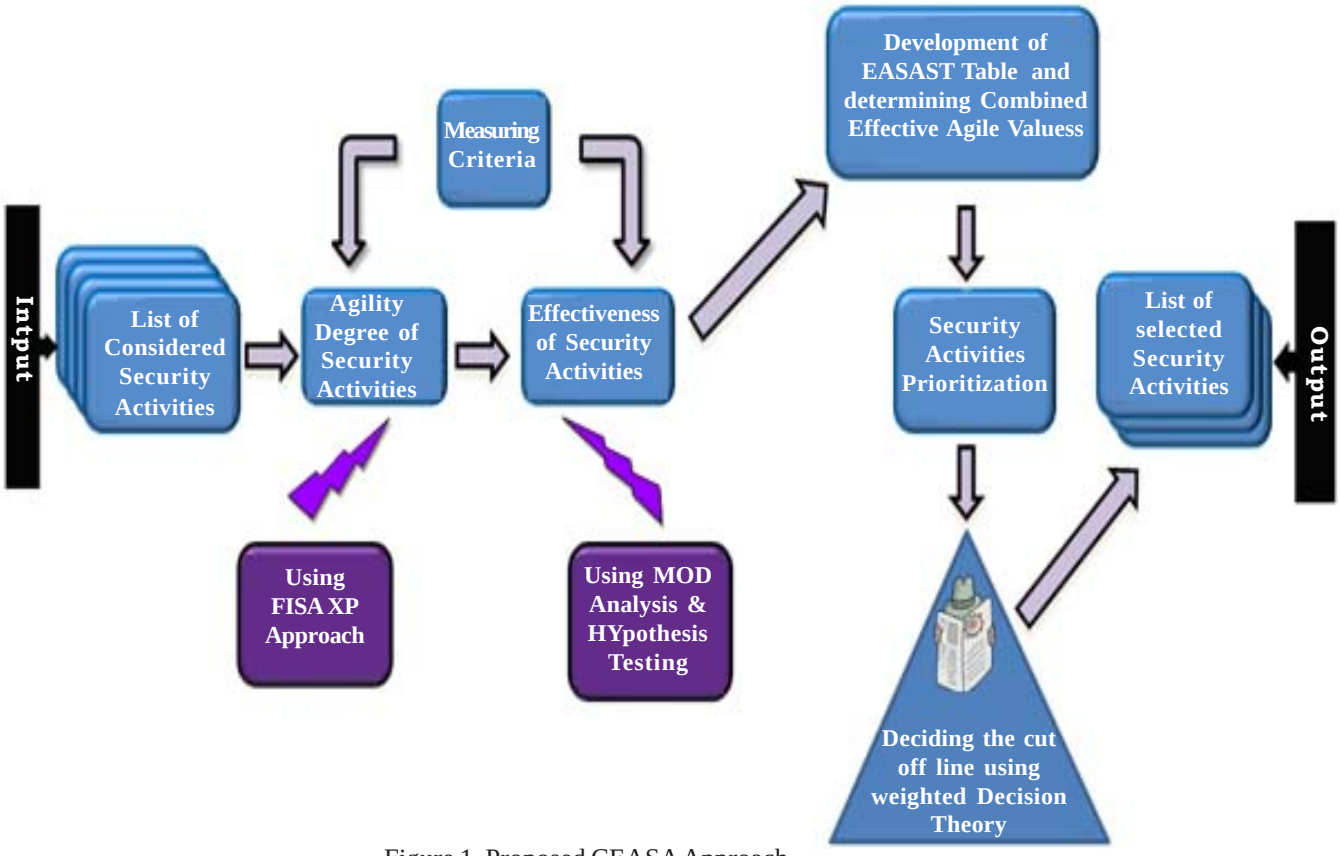

Figure 1. Proposed CEASA Approach

question certain key aspects are considered, named as considerations. Then driver is analyzed on a five-point scale, assisting developers to incorporate different levels of probability in their response. Furthermore, rationale with the consideration is shown using which an analyst can describe the reason for choosing a particular response. This response specifies that up to what extent driver is effective in software security context and hence concludes our approach. Qualitative approach assists developers during secure software development. However, sometimes quantitative evaluation is required to reach to a conclusion. Thus, effectiveness has also been measured using a quantitative analysis. In cases like software engineering where direct knowledge of parameters under consideration is rare, hypothesis testing becomes most suitable. Thus, for quantitative analysis the hypothesis testing approach has been used [3]. For detailed description of these approaches reader can refer [2, 3].

4.3 Step 3: Developing a EASAST (Effective Agile Security Activities Selection Table)

After computing the agility degree and effectiveness in step 1 and step 2, the related security manager or the software developer records all these values in table 1, *EASAST* table.

Development of this table begins by listing all security activities in the first column of the table 1. This can be accomplished with the help of a *FISA-XP* approach [1] described in step 1 of section 5. The agility degree of each security activity is entered in the second column of *EASAST*. For our *CEASA* (Combining Effectiveness and Agility of Security Activity) approach, we have expressed agility degree measures of these security activities on categorical scales since the other measure (effectiveness of security activity) involves only categorical values. To convert agility degree measures into categorical values, we use a simple discretization method named, Equal-width (distance) partitioning. It divides the range into *N* intervals of equal size. Like, if A and B are the lowest and highest values of the attribute, the width of intervals will be

$$W = (B - A) / N$$

Out of many discretization methods available, this method is used as we know that it decides the intervals based on width which will put each security activity in exact interval. In contrast, other methods divide the range into *N* intervals in such a manner that each interval contains approximately same number of samples.

Security Activities	Agility Degree	Effectiveness	CEAV
Perform security analysis of requirements (SA1)	Agile	Extreme	Very High
Specify operational environment (SA2)	Agile	Moderate	High
Identify user roles and requirements (SA3)	Slightly Agile	Moderate	Medium
Detail misuse cases (SA4)	Least Agile	Slight	Low
Identify global security policy (SA5)	Slightly Agile	Least	Low

Table 1. Effective Agile Security Activities Selection Table (EASAST)

The effectiveness of each security activity, determined in step 2, is listed in third column of table 1. Next table 2, *SALT*, defines these measures at the level of detail that is useful for identifying the overall effective agile security activities. Here, we have considered five levels of effectiveness and four levels of agility degree, translating into eight levels of combined effective agile security activities, as shown in table 2.

CEAV, the function of effectiveness and agility rated on an 8-point scale is computed using a Security Activity Lookup Table (*SALT*). In the given table, the ratings support a level of *CEAV* by estimating whether the security activity has a '*Very High*' (7 and 8), '*High*' (5 and 6), 'Medium' (3 and 4), or '*Low*' (1 and 2) *CEAV*. For our considered security activities, the *CEAV* has been calculated using table II and shown in fourth column of table 1.

Agility → Effectiveness ↓	Highly Agile (4)	Agile (3)	Slightly Agile (2)	Least Agile (1)
Extreme (5)	Very High (8)	Very High (7)	High (6)	High (5)
Almost (4)	Very High (7)	High (6)	High (5)	Medium (4)
Moderate (3)	High (6)	High (5)	Medium (4)	Medium (3)
Slight (2)	High (5)	Medium (4)	Medium (3)	Low (2)
Least (1)	Medium (4)	Medium (3)	Low (2)	Low (1)

Table 2. Security Activity Lookup Table (SALT)

4.4 Step 4: Security Activities Prioritization

After assigning the rating to the security activities, the EESAST table is sorted out on the basis of effectiveness and agility measures. Security activities prioritization process is illustrated in table 1 and figure 2 by using five security activities of *CLASP* secure development process, as an example. In table 1, the agility degree, effectiveness and their resulting value *CEAV* has been summarized for the considered security activities. Figure 2 plots effectiveness and agility degree for various security activities considered in table 1. Before plotting this graph, security measures ratings and percent of scales assigned to them have been decided. The two measures and the resultant *CEAV* with their respective ranges considered in our approach have been described in table 3.

From the figure 2 graph and table 1, we can determine that a highly effective and highly agile security activity will have higher rank than an activity with a medium effectiveness and high agility. However, individual values of effectiveness and agility degree are not able to determine the suitability of these security measures. Thus, *CEAV* is an important measure and sorting out of security activities, based on this combined measure, is necessary. Since a '*Highly Agile*' security activity having the '*Least Effectiveness*' reduces the combined value to '*Medium*'. This security activity gets lower rank in the table in comparison to an activity that has a '*Slightly Agile*' and '*Almost*' Effectiveness having high *CEAV* value.

The ranking is required to determine the most important activities for integration. Sorted out security activities with their respective measures are shown in table 1. After sorting out these activities in the table, a cutoff line (horizontal line drawn somewhere in the table) is drawn that means only those security activities which lie above this line will be considered for integration and rest have to be discarded. Cutoff line can be assigned in two different ways. One way to assign this value is from developer's perspective. In this process the top ranked activities comes out using *CEAV* can be selected. Like developer can consider security activities having *CEAV* value more than '*Medium*' or based on certain number that top for activities are selected for integration with agile activity. In another way cutoff line is assigned using a decision taken by client according to

1	**Agility Degree: Percent agility of security provided to the project during integration.**					
	Rating	Highly Agile (4)	Agile (3)	Slightly Agile (2)	Least Agile (1)	
	Percent	70-100	50-70	30-50	0-30	
2	**Effectiveness: Percent of Effectiveness of security activity assist in removing risks when integrated with a project**					
	Rating	Extreme (5)	Almost (4)	Moderate (3)	Slight (2)	Least (1)
	Percent	80-100	60-80	40-60	20-40	0-20
3	**CEAV: Percent of combined Effectiveness & agility of security activity**					
	Rating	Very High (7,8)	High (5,6)	Medium (3,4)	Low (1,2)	
	Percent	70-100	50-70	30-50	0-30	

Table 3. Security Measures and their respective ranges

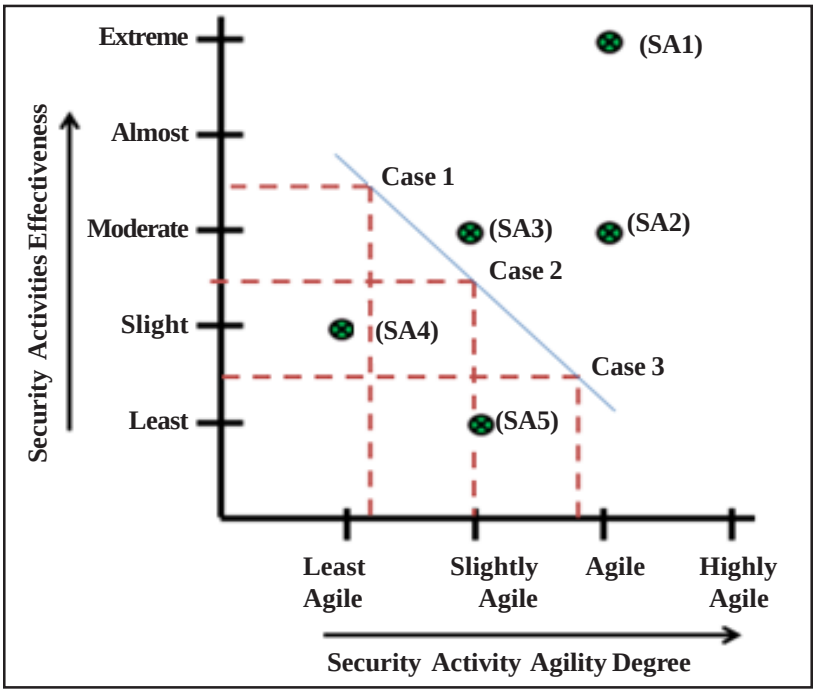

Figure 2. Graph between Agility Degree and Effectiveness of Security Activities

the project's need. The process for deciding cutoff line using client's perspective is described in the next step.

3.5 Step 5: Proper Balancing between two measures using client's perspective

Threats, for which security activities are integrated, are uncertain events that depend on external factors and may change with project type. Thus, it is clear that effectiveness and agility are not equally important for all projects and proper balancing between these two factors must be done.

3.5.1 Assigning weightage to agility and effectiveness

In this step, we will determine the relative suitability of these two factors in a particular project situation, to illustrate the comprehensive impact of effectiveness and agility of security activity during integration.

Three cases for combining these two factors emerge from above discussion as described below.

a) Case 1: This case arises when project's security is more critical aspect and should be given highest priority by the client even though it may decrease the agility. In this case, effectiveness measure of security activity is more important than the agility degree.

i.e. effectiveness weightage > agility degree weightage

Here, during cutoff line decision, developers give 70% weightage to effectiveness of security activity and 30% to agility degree. Moreover, we can't neglect the low priority measure (agility degree) below 30% as we know that for agile projects agility measure is also very important.

b) Case2: In the second case, the client requests that security is important for the project under-development but agility of security activity is equally important. Thus, comparable importance is given to both the measures and equal value is assigned to both of them.

i.e. effectiveness weightage = agility degree weightage

In this case for deciding cutoff line, developers give 50% weightage to effectiveness of security activity and 50% to agility degree. This is the most idealistic selection and can be chosen whenever client is not sure about his decision or even when client is not taking any decision.

c) Case 3: The third case arises when client gives more importance to the agility of security activity for the success of project. This means project security requirements are uncertain, novel and requires flexible development. In this case agility has been given more importance even though security effectiveness may get affected by using it.

i.e. effectiveness weightage < agility degree weightage

In this situation developers can decide cutoff line by giving 30% weightage to effectiveness of security activity and 70% to agility degree. In this security is compromised to a certain extent. However, low priority measure (effectiveness) is also given 30% weightage, thus only those security activities that are less effective (below 30%) are discarded.

3.5.2 Selection from above 3 cases by client using Weighted Decision Theory

The need for getting appropriate option from three cases described above motivated us to focus on client's perspective. To decide which factor is more suitable than the other from client's perspective, we have used a weighted decision theory. According to Maryam Temitayo Ahmed "*Decision making could be defined as the study of identifying and choosing from alternatives, the best option that suits a purpose* [34]". In our approach, we provide certain options to the client. The client can provide more information to the developer by selecting among the options given to him. This makes integration of security activities more meaningful for the project. The options given to client for decision making are briefly described below and listed in table 4.

a) Related to Agility Factors

1. Highly Uncertain and Turbulent Project

Where the project environment is very unstable and requires high flexibility that can accommodate expected or unexpected changes.

2. Project with High Novelty or Responsiveness Requirement

It means that the project involves a new technology or users have never used that software before.

3. Quicker results with maximum support

Company requires early delivery of the working software. For this purpose clients are always available onsite whenever developers need them. Clients provide maximum support and put as much effort as required by developers.

b) Related to Effectiveness Factors

1. Mission Critical Project

It means that project deliverable has no or very low tolerance for defects. In such a situation, it becomes more important to realize that effective security activities need to be given prime importance and then we apply agile features on it.

2. Secure & Robust Software Requirement

This describes a condition where high security is applied in advance by implementing effective security activities before security

violations actually occur. These are required mostly for projects where attacks are frequent.

3. Security becomes Project Standard making cost, time and size secondary
It means that effective security activities are required to be implemented since standard of project is evaluated on the basis of its security level. Accordingly, importance of cost, time and size of project are behind the curtain.

All the options provided to client are given in table 3. We can easily make out that among all these options, three are related to effectiveness measures and other three are related to agility. Weightage for each selection is given as '1' and vice versa. Based on client's selection result, developer decides the best case among the three cases described in Step 5.1. For example, if client selects more agility related options than the 3rd case (Step 5.1) is considered. That is, if total score of agility factors is more than the effectiveness side, then agility is given more importance and vice versa. However, if scores on both the sides are equal then 2nd case of step 5.1 is applicable. Resultant becomes the case which developer opts for deciding the cutoff line. The cutoff line for each case is drawn in graph of figure 2. From this graph we can see that the security activities that lie above the cutoff line will be used for integration with agile activities. Like, we have taken five security activities in our example, which have been given in table I and plotted in the graph of figure 2. In this, if a user has selected case 3 then according to the graph we can see that, security activities 1, 2 and 3 lie above the cutoff line of case 3. Thus these 3 security activities have to be considered for integration with an agile activity and rest should be discarded. Similarly, we can select security activities using figure 2, if any other case is applicable.

What are the characteristics of project? **Which factors are more critical for the success of the project in hand ?**				
Agility Factors	**Score**	**Effectiveness Factors**		**Score**
☐ **Highly Uncertain and Turbulent Project** ☐ **Project with High Novelty or Responsiveness Requirement**		☐ ☐	**Mission Critical Project** **Secure & Robust Software Requirement**	
☐ **Quicker results with maximum support**		☐	**Security becomes Project Standard making cost, time and size secondary**	
Total Score				

Table 4. Weighted Decision Table for proper balancing between two measures

5. Conclusion and Future Work

In the history of software development, a struggle existed between building software by using agile methods or traditional development methods. Agile software development methodologies are now very popular within the companies. They are considered much more efficient than traditional methodologies as they encourage iterative and incremental development with minimum documentation and customer satisfaction. However, organizations still face challenges in the real life scenarios while integrating security activities with agile methods. A security activity relies on security practices that are heavy in nature whereas agile activities are lightweight and use informal methods. Thus, to integrate security activities with agile activities, we must select security activities that are more agile. If a security activity possesses higher agility degree but less effective in removing risks then also it is not beneficial. Focusing on this aspect, this paper presents a *CEASA* approach that combines the agility degree and effectiveness measures of security activities after maintaining proper balance between two. In this approach, first we have measured agility degree and effectiveness of security activities individually and then ranked the security activities based on their combined value. After that, client decides the cutoff line using weighted decision theory keeping in mind the project's need. Finally the security activities that lie above the cutoff line will be considered for integration with agile activities and rest are discarded.

To conclude, it can be said that the use of *CEASA* approach assist software development teams in integrating security in many ways. Using this approach, the software development teams can decide security activities to be integrated for specific project. It takes care of all the requirements of a particular project after keeping agility and security of the project as high as possible. Thus, security activity integration process is not generalized for all projects. Moreover, software development teams should have the confidence and motivation of tailoring the security process to fit in a particular project and according to user's need.

To take this approach further, we may look forward to incorporate more measurement criteria while integrating security activities with agile methods.

References

[1] Sonia, A. Singhal, H. Banati. (2014). FISA-XP: An agile based integration of security activities with Extreme Programming. In ACM SIGSOFT Software Engineering Notes, 39 (3).

[2] Sonia, Singhal, A. (2013). An Evaluation Approach: Measuring Effectiveness of Security Activities. In Seventh International Conference on Data mining and Warehousing (ICDMW), Elsevier.

[3] Sonia, Singhal, A. (2013). Qualitative and Quantitative Evaluation: Measuring Effectiveness of Security Activities. In *IJIP International Journal of Information Processing*, 7 (4), Dec, Bangalore.

[4] Beznosov, K. (2003). 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.

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

[6] https://buildsecurityin.uscert.gov/bsi/articles/bestpractices/requirements/548BSI.html.

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

[8] Ge, X., Paige, R.F., Polack, F., Brooke, P. (2007). Extreme Programming Security Practices. Concas, G. et. al. (eds.) XP, LNCS 4536, p. 226–230. Springer, Heidelberg.

[9] Siponen, M., Baskerville, R., Kuivalainen, T. (2005). Integrating security into agile development methods. *In*: Proceedings of the 38th Annual Hawaii International Conference on System Sciences.

[10] https://www.owasp.org/index.php/Category:OWASP_CLASP_Project.

[11] Baca, D., Carlsson, B. (2011). Agile development with security engineering activities. *In*: Proceedings of the 2011 International Conference on Software and Systems Process ICSSP 2011: 149-158, ACM New York, USA.

[12] Win, B. D., Scandariato, R., Buyens, K., Grégoire, J., Joosen, W. (2009). On the secure software development process: CLASP, SDL and touchpoints compared. *Information and Software Technology*. 51 (7) 1152–1171, July, Pages. Elsevier.

[13] Beznosov, K., Kruchten, P. (2004). Towards Agile Security Assurance. *In*: Proceedings of The New Security Paradigms Workshop,White Point Beach Resort, Nova Scotia, Canada, p. 20-23, September.

[14] Sonia, A. Singhal. (2011). Development of Agile Security Framework using a Hybrid Technique for Requirements Elicitation. In International Conference on Advances in Computing, Communication and Control (ICAC3), Mumbai, India, 125 (1) 178-188.

[15] Sonia,Singhal, A., Banati, H. (2011). Fuzzy Logic Approach for Threat Prioritization in Agile Security Framework using DREAD model. *In International Journal of Computer Science Issues*, 8 (4).

[16] Yang, S. L., Li, T. F. (2002). Agility evaluation of mass customization product manufacturing, *Journal of Materials Processing Technology*, 129 (1–3) 640–644.

[17] Ren, J., Yusuf, Y. Y., Burns, N. D. (2000). A prototype of measurement system for agile enterprise. *In*: International Conference on Quality, Reliability, and Maintenance. Oxford, UK, p. 247–252.

[18] Ren, J., Yusuf, Y.Y., Burns, N. D. (2009). A decision-support framework for agile enterprise partnering. *In*: *The International Journal of Advanced Manufacturing Technology*, March, 41 (1-2) 180-192.

[19] Qumer, A., Henderson-Sellers, B. (2008). An evaluation of the degree of agility in six agile methods and its applicability for method engineering. Information and Software Technology, 50 (2008), Elsevier, p. 280-295

[20] Sherehiy, B., Karwowski, W., Layer, J. K. (2007). A review of enterprise agility: Concepts, frameworks, and attributes. International Journal of industrial ergonomics, 37, Elsevier, p. 445-460.

[21] Al-Ahmad, W., Building Secure Software using XP. *International Journal of Secure Software Engineering* (IJSSE), 2 (3).

[22] Sarker, S., Munson, C. L., Sarker, S., Chakraborty, S. (2009). Assessing relative contribution of facets of agility to distributed systems development success: An analytic hierarchy process approach. *European Journal of Information Systems*, 18 (4) 285-299.

[23] Sonia, A. Singhal, H. Banati. (2013). Measuring Relative Importance of Agility Features Contributing Towards Agility of a Software Process. *In*: Fifth International Conference on Advances in Recent Technologies in Communication and Computing, Bangalore, India, Elsevier.

[24] Keramati, H., Hassan, S., Hosseinabadi, M. (2008). Integrating software development security activities with agile methodologies. IEEE/ACS International Conference on Computer Systems & Applications.

[25] Sonia, Singhal, A. (2012). Integration Analysis of Security Activities from the perspective of agility. *In*: International Conference on Agile and Lean software methods (AI), February 17 -19, Bengaluru, India.

[26] Alberts, C., Allen, J., Stoddard, R. (2012). Risk-Based Measurement and Analysis: Application to Software Security (CMU/SEI-2012-TN-004). *Software Engineering Institute*, Carnegie Mellon University.

[27] Butler, S. M. (2002). Security Attribute Evaluation Method: A cost-benefit approach. *In*: Proceedings of the 24th International Conference on Software Engineering, Orlando, Florida, USA, May 19 - 25.

[28] Bartol, N., Hamilton, B.A. (2008). Practical Measurement Framework for Software Assurance and Information Security. Practical Software and Systems Measurement Support Center.

[29] Alberts, C., Allen, J., Stoddard, R. (2010). Integrated Measurement and Analysis Framework for Software Security (CMU/SEI-2010-TN-025). Software Engineering Institute, Carnegie Mellon University.

[30] Ayalew, Eyader T., Kidne, Abreham, T. (2012). Identification and evaluation of security activities in agile projects: A systematic Literature Review and Survey study, Blekinge Institute of Technology, September.

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

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

[33] Kurian,Tisni. (2005). Agility in IT: Creating Software through Agile Techniques. *In*: Proceedings of the IEI-International Conference on Agility Design & Manufacturing, ADEMSE-Bangalore, India.

[34] Temitayo Ahmed, M., Omotunde, H. (2012). Theories And Strategies of Good Decision Making, *In*: *International Journal Of Scientific & Technology Research*, 1 (10), November.