Journal of Theoretical and Applied Information Technology
15th August 2017. Vol.95. No.15

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

ELICITING SECURITY REQUIRMENTS FOR MOBILE APPS: A REPLICATION STUDY

[1]NOORREZAM YUSOP, [2]MASSILA KAMALRUDIN, [2]MOKHTAR MOHD YUSOF, [2]SAFIAH SIDEK

[1] Faculty of Communication and Information Technology, Universiti Teknikal Malaysia Melaka, MALAYSIA

[2] Innovative Software System and Services Group, Universiti Teknikal Malaysia Melaka, MALAYSIA

E-mail: [1]p031320001@student.utem.edu.my, [2]{massila, mokhtaryusof, safiahsidek}@utem.edu.my

ABSTRACT

Mobile applications (mobile apps) are becoming a common medium for conducting transaction, saving data and exchanging information online. However, an important issue that has been overlooked is the emphasis on security issues at the early stage of mobile apps development. It has become a common practice among requirements engineers to deal with security issues after the mobile apps have been developed. This scenario has led to the failure of developing secure and safe mobile application based on the needs of the users. Motivated by this problem, we propose an automated support tool to assist requirements engineers to elicit security related requirements at the early stage of mobile apps development. This paper reported a replication of a study from our previous work that describes our user study and tool support, called MobiMEReq. This tool uses SecEUCs and SecEUIs prototype model to automatically elicit the security attributes requirements of mobile apps. In this paper, we reported the results drawn from an experiment of a user study to compare the capability of the MobiMEReq in relation to the manual approach. The results of the user study show that the tool support has higher accuracy rate in comparison to the manual approach to extract security attributes elicited from functional requirements. This implies that our tool is able to help requirements engineers to easily elicit security attribute requirements of mobile apps.

Keywords: *Security requirements, Security attributes, Mobile apps, Security requirements elicitation*

1. INTRODUCTION

Mobile applications (mobile apps) are becoming a common medium for conducting transaction, saving data and exchanging information online. However, an important issue that has increasingly become a concern is the lack of emphasis on security issues at the early stage of mobile apps development. Issues related to securities are commonly being dealt with at the later stage of developing the mobile apps. Further, it has been a frequent practice among requirements engineers to ignore or incorrectly elicit security-related requirements during the early stage of mobile apps development. This practice, if not tackled may lead to the failure of developing a secure and safe mobile application.

There are several reasons why this issue needs to be addressed. Firstly, there are possibilities that the requirements engineers fail to elicit correct security requirements while conducting the elicitation because they may face difficulties to understand the terms and knowledge of the security [1]. Secondly, the quality of software development is highly dependable on the process of capturing correct and consistent requirements from client-stakeholders. However, this process is often difficult, time consuming and error prone [2][3]. Motivated by this problem, we propose an automated support tool to assist requirements engineers to elicit security related requirements at the early stage of mobile apps development.

We believe that the automated support tool for eliciting security related requirements at the early stage of mobile apps development is crucial due to the following reasons. Firstly, the complexity of the Common Criteria (CC) of the security requirements makes it difficult to understand, especially the novice requirements engineers [3]. CC describes the requirements in two categories: 1) the functional requirements, and 2) the assurance requirements. In security behavior, the CC is

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

described in both types [4]. Developers tend to make mistakes when determining the right security requirements and attributes because they need to identify the requirements and attributes personally without any supports, such as the automation or the manual training. Secondly, there is no predefined instruction provided to the user when using the GUI for dynamic analysis. This leads to various challenges in completing the security identification process [5][6][7]. The aforementioned scenario indicates the need for an automation that can help to elicit security requirements and attributes, especially for novice requirements engineers.

Several approaches have been proposed to tackle the problems mentioned. For example, Haley et al. [8] proposed an approach to support security requirements elicitation and analysis. They proposed a method to construct a system context using a problem-oriented notation. However, due to the complexity of the proposed approach, they require experts to construct the setting and analysis.

Another approach has been proposed by El-Hadary et al.[9]. They proposed a method to capture security requirements for software systems. The method allows for early integration of security requirements with software development using problem frames. It also identifies security requirements with the aid of previous knowledge through the construction of security catalogue [9]. However, the proposed method is limited to certain domain categories and does not elicit security requirements for security attributes.

Highlighting the importance of security knowledge, Berger et al. [10] claimed that software engineers lack the security knowledge although this body of knowledge is easily accessible. They argued that both the software engineers and developers have problems in selecting the relevant piece of security knowledge and they have difficulties to extract and make decision for their design or requirements.

Studies to reuse security knowledge to assist software developers in eliciting security requirements in a systematic way have been conducted by using different approaches, such as security problem frames [11], misuse cases templates [12], and anti-models patterns [13]. These approaches are used to form generic model based on catalogues not specified for a particular application. Thus, the developer can reuse such generic models and templates [14][15].

Our study was a replication of experiments conducted in a previous study [16] on eliciting security attributes to assess the ability and coverage of our tool approach. Similar to our previous work [17], a user study was conducted to gauge the ability of the requirements engineers to elicit the security related requirements from a set of business requirements of a mobile app.

This paper describes a proactive approach of a tool support that automatically elicits security requirements of mobile apps using Essential Use Cases (EUCs) and Essential User Interface (EUI) prototype models as well as a replication of study from [16] in a different study. In this paper, firstly, we describe the background of the study. Secondly, we present the research methodology of our user study. Next, we describe the results of the experiment that compares the performance of the tool in eliciting security attributes to the same requirements samples as per discussed in [17]. Further, we discuss a study that aims to prove its correctness in eliciting a range of security attributes from several sets of security requirements. Next, we describe the validity of experiments result. Finally, we discuss the implications of these studies and the prototype as well as our future work.

Based on our earlier finding [16], that engineers are poor in eliciting correct security requirements. Hence, this study is conducted to investigate further problem with different set of respondents in order to gain consistent findings.

This study is aimed to answer the following:

1. Can replication study help to elicit security requirements is better than manual approach?

2. Does the replication of study is help to overcome the issues elicitation?

3. How the replication of study use for target usefulness of tool evaluation?

2. BACKGROUND AND MOTIVATION

2.1 Security requirements attributes

Security requirements attribute as well as security attribute can be defined as any piece of information that may be associated with a controlled implicit entity or user for the purpose of implementing a security policy. However, it is not necessarily be implemented directly in data structures [18]. Figure 1 (A) [19] describes the security related for security requirements and Figure 1 (B) describes the attributes used for each security related requirements.

Journal of Theoretical and Applied Information Technology
15th August 2017. Vol.95. No.15

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

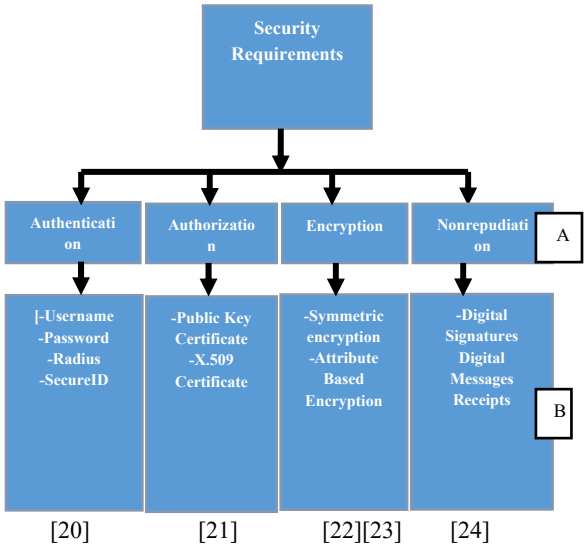

Figure 1: Security Requirements And Its Related Security Attributes

In this study, security requirements address the security issues at the early stage of system design, while accommodating the complex needs of different stakeholders. Based on our previous work [25], the security requirements were found to be similar for mobile application development, and they are normally considered at the later phase of the system or mobile apps development.

2.2 Essential Use Case (EUC) and Essential User Interface (EUI)

The EUC approach, sometimes called as a business use case has been defined by Constantine and Lockwood as a "structured narrative, expressed in a language of the application domain and of users, comprising a simplified, generalized, abstract, technology free and independent description of one task or interaction that is complete, meaningful, and well-defined from the point of view of users in a role or some roles in relation to a system, and that embodies the purpose or intentions underlying the interaction" [26]. Biddle defined that the main objectives of EUC is to support better communication between the developers and stakeholders via a technology-free model and to assist better requirements capture. This allows for the capture of specific details relevant for the intended design [27]. Figure 2 shows the example of natural language requirements (lefthand side) and EUC (right hand side) when capturing the requirements (adapted from [26]). The natural language requirements from which the important phrases are extracted (highlighted in yellow) are shown on the left hand side of Figure 2.

EUI prototyping is a low fidelity prototyping approach [28]. It supplies a general idea that corresponds to UI, but does not supply the full detail of UI. Further, it focuses on the requirements rather than the design, representing UI requirements without the need for prototyping tools or widgets to draw the UI [29]. EUI prototyping extends from and works in tandem with the semi-formal representation of EUCs. By focusing on the users and their usage of the system, rather than the system features [30], it helps clients and the requirements engineers to avoid from being misled or confused by chaotic, rapidly evolving and distracting details. Figure 3 shows the example of EUI prototype, comprising the EUCs.

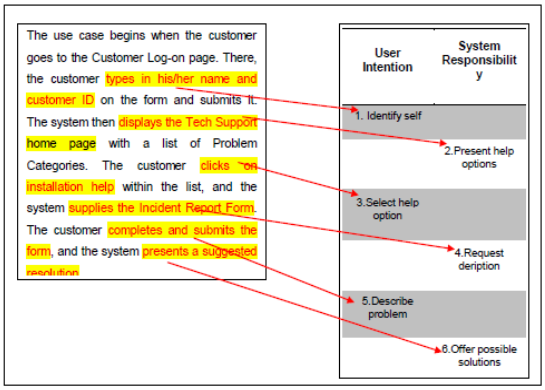

Figure 2: Example Of Textual Natural Language Requirements (Left) And Example Of Essential Use Case (EUC Model) (Right)

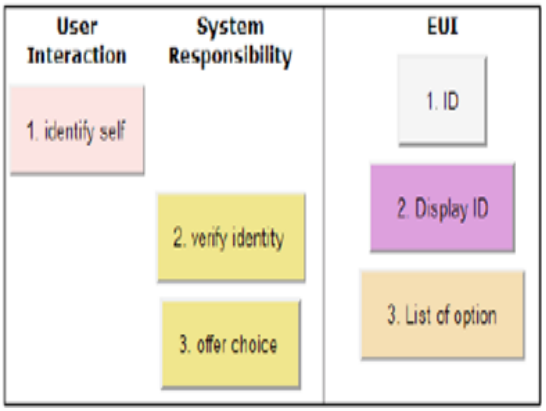

Figure 3: Example Of EUI Prototype From EUC Models

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

2.3 Security Essential Use Case (SecEUC) and Security Essential User Interface (SecEUI)

SecEUC is a security pattern library that comprises security related EUC, while SecEUI is the security related EUI. A collection of SecEUCs has been defined by Yahya et al. [31], in which they developed a security pattern library that comprises security related EUC, known as the SecEUCs and security related essential interaction identified as the SecEI pattern library. Examples of the SecEI and the SecEUC are shown in Table 1. They used EUC model to capture security requirements from business requirements to allow requirements engineers to identify and capture the security requirements consistently.

For the purpose of this study, we chose the SecEUC, SecAttributes and Mobile Security pattern library as well as their model in order to conduct an in-depth analysis that could help to capture and validate security requirements from business requirements and mobile apps. The SecEUC library patterns, based on EUCs were generated from normal business requirements, while the SecEI library patterns were based on the essential interactions found in the textual requirements related to security elements. The development of SecEUC pattern library was an adaptation from the works by [32][33][34][35]. This approach has led to the identification of associated security elements that were based on the definitions from the basic security services. The Mobile security pattern [36] was used to support the extraction of the security related attributes [16] from the security requirements.

The patterns are generic and could support different domain of application.

Table 1: Example Of Seceuc Pattern Libraries [28].

SecEI	SecEUC	SecCtrl
Check username	Identify self	Authentication
Check password		
Verify username		
Make payment	Make payment	Transaction
Complete payment form		

3. STUDY METHOD

This section describes the design of our user study. Aiming to investigate the ability of the tool to assist requirements engineers to elicit correct security attributes from business security requirements, this study is a replication of our previous work, which also has a similar aim.

Case study was used in comparison to the previous [16], but with the same level of complexity. This level of complexity is verified by an expert. Further explanation is as follows:

3.1 Subject of the Study

The subjects of this study were 50 third-year students of a public university in Malaysia, namely the Universiti Teknikal Malaysia Melaka (UTeM). At the time of the study, these students were enrolled in a course of software testing and quality assurance.

Prior to the study, they were given a written informed consent form, and all of them agreed and volunteered to participate in this experiment. The participant were informed that: (i) the experiment is not mandatory, (ii) they will be observed while performing the task, (iii) they were not evaluated on their performance and (iv) data collected will be used only for research purposes.

3.2 Study Materials

The study materials consisted of a tutorial and a set of security requirements sample. The tutorial explained the SecEUC and SecEUI model that are used as the requirements model in this experiment. Participants were also provided with a requirements sample for Mobile i-Health apps in the form of use case scenario as given below.

"This mobile application named i-health apps could support mobile online application based on patient health monitoring. I-health provides highly secure information for patient. For this scenario, patient must register their information as a member so that the system can allow patient to access the application. Patient will login to i-health to view patient information and details such as the username, password, identity card no, phone no and email from their console. Patient also can choose the menu selection option to view the patient medical record exercise. Patient is compulsory to do exercise 3 times a week based on the exercise classification and all of these need to be recorded in the patient exercise console provided. Patient can modify certain information related to the mobile apps. This i-health provides automatic notification to patient with incomplete

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

exercise. It also provides patient monthly summary report for doctor record."

The associated SecEUC model derived from the sample requirements as shown in Table 2 is also given to the participants.

Table 2: The Seceuc Generated From I-Health Apps Security Requirements Scenario

Functional Req.	SecEUC
Register	Form
Login	Identify Self
Menu Option	Option
Patient record	View Record
Notification	Alert
Monthly Report	Report

3.3 Variable Selection

We also identified the independent and dependent variables relevant to our study. The dependent variable provides some sort of behavior or response [37]. The dependent variables of our study are i) the participant's comprehension level, and ii) time taken to elicit security attributes from the SecEUC model. The comprehension level is to measure the participants' ability and skills. The comprehension level was measured by checking the correctness of the elicitation security attributes to the SecEUC model. Additionally, time was used to measure the efforts required to elicit security attributes from the security requirements model.

Zowghi and Gervasi [38] have suggested that correctness has at least two different perspectives: i) the formal point of view, correctness is usually meant to be a combination of consistency and completeness. Consistency refers to a situation where a specification contains no internal contradictions whereas completeness refers to a specification that entails everything that is known to be true in a certain context; ii) the practical point of view, correctness refers to the satisfaction of business goals. This indeed is the kind of correctness which is more relevant to the customer, whose goal is to have a newly developed system to meet his overall business goal. Our focus in this study is the formal point of view, which is the combination of consistency and completeness of the security attributes.

Table 3 displays our correctness measurements in this experimentation. As described in Table 3, we have two security attributes for the login procedure in our tool tracing from the pattern library. We defined that correct answer and wrong answer given by the participants. Specifically, the participant has a correct security attributes when he or she generates similar security attributes from our pattern library. Meanwhile, a participant's response is considered as wrong security attributes if none of the defined security attributes matches with our pattern library.

3.4 Experiment Procedure

The experiment was conducted during one of the teaching and learning sessions in a computer lab. The main task in the experiment is to request the participants to manually elicit the security attributes from EUC model. Prior to that, they were given a short description of the conduct of the experiment. We also provided a tutorial session that gives the participants the theory of SecEUC model in detail and an example on the process of eliciting security attributes from the model. They were given 20 minutes to understand the concept and some hands-on examples during the tutorial session. Then, the participants were requested to attempt the following tasks. Before the experiment, the participants are requested to: i) Read the sample on i-health mobile apps business requirements for 5 minutes; and b) write their matric card number on the sheet given.

Further, during the experiment, we informed the participants the specific time to start the task. They are expected to: i) Write the security attributes on the provided sheets; and ii) Once they have completed the task, write down the specific end- time and call the researcher.

4. RESULTS

4.1 User study: Manual vs. Automatic Extraction

We compared the correctness and the performance of the tool with the manual extraction of the requirements by 50 novices as describes. Based on the result shown in Table 3, the automated tool produced 90% correctness in comparison to only to 42% correctness from the manual approach, as reported in [25]. In comparison to our previous work [16] it is found that the result of perform in term of correctness and time taken are nearly similar as per Table 4. This implies that our tool automated has the ability to elicit an almost correct security attributes than the manual approach. However, the 10% errors made by the automated tool were its failure to capture the "PatientId" for register requirements, "ExerciseId" in patient record requirements and "ContactNo" in monthly report. It is believed that the failure of the

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

tool to capture these requirements is due to the passive structure phrases written in the requirement. Based on the high percentage of correctness demonstrated by the tool, we can conclude that MobiMEReq is able to facilitate the participants to capture correct security attributes. Further, the time taken to execute the extraction process only took 1 second in comparison to the manual approach which took approximately 30 minutes to extract the security attributes by the participant in [17].

Table 3: Correctness Between Manual Extraction And Automated Validating Tool

Functional Requirement	Answers	No. Correct answers		No. Wrong answers	
		Manual	Tool Tracing	Manual	Tool Tracing
Register	Username	46	50	4	0
	Password	46	50	4	0
	Email	45	50	5	0
	PatientId	47	49	3	1
	ContactNo	45	50	5	0
Login	Username	48	50	2	0
	Password	48	50	2	0
Menu Option	MenuId	14	50	36	0
	Username	14	50	36	0
	Password	8	50	42	0
Patient record	Email	22	50	28	0
	PatientId	25	50	25	0
	ContactNo	20	50	30	0
	RecordId	10	50	40	0
	ExerciseId	10	47	40	3
	Username	32	50	18	0
	Password	32	50	18	0
Notification	NotificationId	3	50	47	0
	RecordId	5	50	45	0
	PatientId	9	50	41	0
	Username	8	50	42	0
	Password	8	50	42	0
Monthly Report	StaffId	12	50	7	0
	PatientId	14	50	3	0
	ReportId	9	50	41	0
	Email	6	50	44	0
	ContactNo	6	48	44	2
	Username	15	50	35	0
	Password	14	50	36	0
Correctness ratio		621	1394	765	6
		42%	90%	58%	10%

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

Table 4: Comparison Between Experiment 1 [16] And Experiment 2 Correctness Between Manual Extraction And Automated Validating Tool

Experiment	Experiment 1 [16]	Experiment 2
Correctness ratio	46%	42%
Tool tracing	95%	90%
Time taken	25 minutes	30 minutes

5. DISCUSSION

Based on our observation during the conduct of the experiment and the comparison analysis between the correctness of the automated validating tool and the manual extraction of the previous, it was found that the automated tool facilitated the participants to extract almost more than double correct security attributes in comparison to the manual extraction. Specifically, the automated extraction process took just over 1 second to execute in comparison to the average duration of half hours taken to extract the security manually by the participants. The accuracy of the manual elicitation is 42%, while the accuracy of the automated elicitation facilitated by MobiMEReq is 90%. On the other hand, the percentage of incorrect extraction security attributes from the MobiMEReq is lower than the manual approach, which is 10% in comparison to 58% respectively.

In summary, these results indicate that MobiMEReq embedded with the SecAttributes pattern library [16] has higher accuracy rate in comparison to the manual approach to extract security attributes elicited from functional requirements. Based on the result, it is proven that the tool is useful in helping novice requirements engineers and software developers from different background to extract security attributes. The result for this experiment shows that generally, the participants agreed that the tool is helping them to elicit correct security attributes from the functional requirements.

The result on this study and previous work is significant which both of work is agreed that the tool is helping them to elicit correct security attributes from the functional requirements.

6. VALIDITY OF RESULTS EXPERIMENTS

In this section, we discuss the validity of the experimental results to highlight the limitations and strengths of the study. There are two distinct forms of validity, which researchers are concerned about when using experimentation, namely the internal and external validity [39].

Internal validity measures the cause-effect relationship identified in a study [40]. Examples of internal validity are history, pre-testing, maturation, instrumentation, sampling bias and mortality. For the purpose of our study, the historical effect was addressed by ensuring that all participants conducted the experiment at the same time and place. For the pre-test effect, we purposely made sure that the all the participants were properly trained and given sufficient theoretical knowledge before they begin the experiment as described in Section 3. They were also not aware of the main objective of experimentation. With respect to maturity effect, we made sure that the participants were clearly informed that their response will be treated anonymously and they were not evaluated on their performance. This was achieved by asking them to read and sign the consent forms. With respect to instrumentation effect, the participants were recalibrated by using the questionnaires as measurement instrument for consistency. Additionally, the results of the participants were not compared to one another because they differ in some important aspects. Finally to address the sampling bias, we made sure that participants are all students who enrolled in the same course.

External validity refers to the degree to which the results of an empirical investigation can be generalized to and across individuals, settings and times [40] and its confounding are interactive effects of testing, interactive effect of sampling bias and contrived situations [39]. To address the interactive effects of testing, we identified that the participants were considered as novice software engineers with an approximately equal knowledge, hence they may not be well-trained or be well-trained. By the same measure, questionnaire may pose the questions in a different way, the participant differ to understand. With respect to Interactive effect of sampling bias, all participants attempted the same task. Further, the main task of the experiment is to elicit security attributes and the participants were not well trained in this area. In this case, the sample of the security requirements give to the participants was not complex, hence, this measure helped to address the contrived situation.

Thus, there are several positive and negative validity result to our study. To tackle the threats of validity, the tutorial video on this study is needed to relate to the understanding of requirement engineer to elicit manually and automated generated elicitation security attributes

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

in order to reduce the complexity and the time taken to complete the task.

The limitation and assumption undertaken of this study during evaluation is: (Limitation) you use only set of requirements and the results might be different if you use more. (Assumption) student knowledge level is similar with novice requirements engineer.

7. CONCLUSION AND FUTURE WORK

The growth in mobile devices and package of mobile apps has been an important medium to conduct transaction, saving data and exchange information online. In this case, each of the existing mobile apps and approaches is useful when elicit security requirements. However, there is a lack of emphasis on security issues during the development of mobile apps. Therefore, the need to provide correct security attributes to capture security requirements from client stakeholders is one of the important goals to improve security requirements elicitation. For this purpose, we proposed an automated support tool to assist requirements engineers to elicit security related requirements at the early stage of mobile apps development. The tool support is called MobiMEReq for security requirements of mobile apps by using SecEUCs and SecEUIs prototype model in our work [16]. In this paper, we reported the results drawn from an experiment of a user study to compare the capability of the tools in relation to the manual approach. The results of the user study indicate that the tool support MobiMEReq is able to automatically elicit the security attributes of mobile apps. Our future work is to provide the end-to-end validation approach that can capture and validate security requirements for mobile apps.

ACKNOWLEDGEMENT:

We would like to thank Universiti Teknikal Malaysia Melaka for its support and FRGS grant: FRGS/1/2016/ICT01/FTMK-CACT/F00325 and also Ministry of Higher Education (MOHE), MyBrain15.

[1] K. Schneider, E. Knauss, S. Houmb, S. Islam, and J. Jürjens, "Enhancing security requirements engineering by organizational learning", *Requir. Eng.*, vol. 17, no. 1, pp. 35–56, 2012.

[2] M. Kamalrudin and J. Grundy, "Generating Essential User Interface Prototypes to Validate Requirements," 2011, pp. 564–567.

[3] E. Paja, F. Dalpiaz, M. Poggianella, P. Roberti, and P. Giorgini, "STS-Tool : Socio-Technical Security Requirements through Social Commitments", 2012.

[4] M. S. Ware, J. B. Bowles, and C. M. Eastman, "Using the Common Criteria to Elicit Security Requirements with Use Cases," pp. 273–278, 2006.

[5] P. Aho and T. Räty, "Enhancing Generated Java GUI Models with Valid Test Data", in *2011 IEEE Conference on Open Systems (ICOS2011), September 25-28 2011, Langkawi , Malaysia*, 2011, pp. 310–315.

[6] A. Kull, "Automatic GUI Model Generation : State of the Art", 2012.

[7] N. Yusop, M. Kamalrudin, S. Sakinah, and S. Sidek, "VALIDATION OF SECURITY REQUIREMENTS FOR MOBILE APPLICATION : A STUDY", *Sci. Int.*, vol. 2014, no. October, pp. 1451–1454, 2014.

[8] C. B. Haley, R. Laney, and J. D. Moffett, "Security Requirements Engineering : A Framework for Representation and Analysis", *IEEE Trans. Softw. Eng.*, vol. 34, no. 1, pp. 133–153, 2008.

[9] H. El-hadary and S. El-kassas, "Capturing security requirements for software systems", *J. Adv. Res.*, vol. 5, no. 4, pp. 463–472, 2014.

[10] B. J. Berger, K. Sohr, and R. Koschke, "Extracting and Analyzing the Implemented Security Architecture of Business Applications", in *Proceeding of the 17th European Conference on Software Maintenance and Reengineering*, 2013.

[11] D. Hatebur, M. Heisel, and H. Schmidt, "A Pattern System for Security Requirements Engineering", in *The Second International Conference on Availability, Reliability and Security, 2007. ARES 2007*, 2007, pp. 356–365.

[12] G. Sindre, D. G. Firesmith, and A. L. Opdahl, "A Reuse-Based Approach to Determining Security Requirements", in *Proceeding of the 9th international workshop on requirements engineering: foundation for software quality (REFSQ'03)*, 2003.

[13] L. A. Hermoye, A. Van Lamsweerde, and D. E. Perry, "A Reuse-Based Approach to Security Requirements Engineering," in *In Proc. 9th International Workshop on Requirements Engineering: Foundation for Software Quality (REFSQ'03*, 2003.

[14] M. Kamalrudin, S. Sidek, M. N. Aiza, and J. Grundy, "AUTOMATED ACCEPTANCE TESTING TOOLS EVALUATION IN

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

AGILE SOFTWARE DEVELOPMENT", in *Science International (Lahore)*, 2013, pp. 1053–1058.

[15] M. Kamalrudin, M. N. Aiza, J. Grundy, J. Hosking, and M. Robinson, "Automatic Acceptance Test Case Generation From Essential Use Cases", in *13th International Conference on Intelligent Software Methodologies, Tools and Techniques (SOMET), Langkawi, Malaysia, September 22-24, 2014*, 2014.

[16] N. Yusop, M. Kamalrudin, and S. Sidek, "CAPTURING SECURITY REQUIREMENTS OF MOBILE APPS USING MobiMEReq Noorrezam Yusop", *Asia Pacific J. Contemp. Educ. Commun. Technol.*, vol. 3, no. 1, 2017.

[17] N. Yusop, M. Kamalrudin, M. Mohd Yusof, and S. Sidek, "Meeting Real Challenges in Eliciting Security Attributes for Mobile Application Development", *Journal. Internet Computing and Services.*, vol. 0170, no. 5, pp. 25–32, 2016.

[18] I. Krka, G. Edwards, L. Cheung, L. Golubchik, and N. Medvidovic, "A comprehensive exploration of challenges in architecture-based reliability estimation," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*,, 1973, p. A comprehensive exploration of.

[19] G. Ian, *Essential Software Architecture*. 2006, pp. 1–283.

[20] "User Authentication in Mobile", [Online]. Available: https://sc1.checkpoint.com/documents/R77/CP_R77_Mobile_Access_WebAdmin/41587.htm.

[21] P. Vilhan and L. Hudec, "Building Public Key Infrastructure for MANET with Help of B.A.T.M.A.N. Advanced", in *Proceeding of the Modelling Symposium (EMS), 2013 European, Manchester, 20-22 Nov*, 2013, pp. 566–571.

[22] A. Rekha, P. Anitha, A. . Subaira, and C. Vinothini, "A Survey on Encryption Algorithms for Data Security", *Int. J. Res. Eng. Technol.*, pp. 131–134, 2015.

[23] C. Loftis, T. Chen, and J. M. Cirella, "Attribute-level encryption of data in public Android databases", *(RTI Press publication OP-0016-1309, Research Triangle Park, NC: RTI Press*, 2013.

[24] C. . Chen and W. . Tsai, "Using a Stored-Value Card to Provide an Added-Value Service of Payment Protocol in VANET", in *Proceeding of the Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS),2013 Seventh International Conference, 3-5 July 2013, Taichun*, 2013, pp. 660–665.

[25] N. Yusop, M. Kamalrudin, and S. Sidek, "Jurnal Teknologi SECURITY REQUIREMENTS VALIDATION FOR MOBILE APPS : A SYSTEMATIC LITERATURE REVIEW", *Journal. Teknologi.*, vol. 34, pp. 123–137, 2015.

[26] L. L. Constantine and L. A. D. Lockwood, *Software for use: a practical guide to the models and methods of usage-centered design*. Pearson Education (1999), 1999.

[27] R. Biddle, J. Noble, and E. Tempero, "Essential Use Cases and Responsibility in Object-Oriented Development", *ACSC '02 Proc. twenty-fifth Australas. Conf. Comput. Sci.*, vol. 3, no. 1, pp. 7–16, 2002.

[28] S. W. Ambler, "Essential (Low Fidelity) User Interface Prototypes", 2003. [Online]. Available: http://www.agilemodeling.com/artifacts/essentialUI.htm.

[29] L. L. Constantine and L. A. Lockwood, "Usage-centered software engineering: an agile approach to integrating users, user interfaces, and usability into software engineering practice", in *Proceeding of the 25th International Conference on Software Engineering (ICSE'03) 2003, IEEE Computer Society, Portland, Oregon*, 2003.

[30] S. W. Ambler, "The Object Primer: Agile Model-Driven Development with UML 2.0 (3rd ed.) ", 2004.

[31] S. Yahya, M. Kamalrudin, S. Sidek, and J. Grundy, "Capturing Security Requirements Using Essential Use Cases (EUCs) ", in *Asia Pacific Requirements Engineering Symposium (APRES) 2014*, 2014, pp. 16–30.

[32] M. Kamalrudin, J. Grundy, and J. Hosking, "Tool Support for Essential Use Cases to Better Capture Software Requirements", in *Proceeding of the of IEEE/ACM international conference on Automated software engineering*, 2010, pp. 327–336.

[33] M. Kamalrudin, "Automated Software Tool Support for Checking the Inconsistency of Requirements", in *Proceeding of the 24th IEEE/ACM International Conference on Automated Software Engineerin*, 2009.

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

[34] M. Kamalrudin, J. Grundy, and J. Hosking, "Automated Support for Consistency Management and Validation of Requirements", 2011.
[35] M. Kamalrudin and S. Sidek, "A review on software requirements validation and consistency management", *Int. J. Softw. Eng. Its Appl.*, 2015.
[36] N. Yusop, M. Kamalrudin, S. Sidek, and J. Grundy, "Automated Support to Capture and Validate Security Requirements for Mobile Apps", in *Asia Pacific Requirements Engineering Symposium (APRES)*, 2016, no. November, pp. 10–12.
[37] "Designing an Experiment: The Variable and the Groups" [Online]. Available: http://www.tulsa.oklahoma.net/~jnichols/Experiment.html.
[38] D. Zowghi, "On the Interplay Between Consistency , Completeness , and Correctness in Requirements Evolution", *Elsevier Sci.*, no. April 2003, pp. 1–37, 2003.
[39] I. . Crawford, "Marketing Research and Information Systems", 1997.
[40] R. K. Yin, *Case Study Research: Design and Methods Fourth Edition*, vol. 5. 2009.