ISSN:2231-0711

Integrated Framework of Software Engineering and Common Criteria Practices

Khairunnisa Osman

[#]*Information Security Certification Body (ISCB),CyberSecurity Malaysia, Level 7, SAPURA@MINES, 7, Jalan Tasik, The Mines Resort City, 43300 Seri Kembangan, Selangor Darul Ehsan, Malaysia*

***Abstract*— Software developers are usually familiar with software engineering deliverables but face difficulties in providing the deliverables that must be in line with the Common Criteria requirement. The software engineering deliverables lack the security requirements to be the evidences in the Common Criteria evaluation and certification. Therefore, the main aim is to develop a framework between Common Criteria and software engineering deliverables. This project objective are to investigate the practices of software engineering and Common Criteria, consolidate the deliverables between software engineering and Common Criteria and solicit an evaluation of the integrated framework design by the developers of the software, evaluators and certifiers of Common Criteria. The investigation on the software engineering practices using the technique of Systematic Literature Review has been conducted and it was found that the ISO/IEC 12207:2008 as the latest standard practices among software developers in developing software. The consolidation used Causal, Semantic and Concept mapping between the process of Software Engineering and Common Criteria to see the similarities between both processes and deliverables before being integrated into the framework. The development of the framework was conducted after the similarities between the processes and deliverables of Software Engineering and Common Criteria are found. The evaluation used a questionnaire that was distributed among experts in Common Criteria and Software Engineering and it found that the framework gives a perceived ease of adoption and less apprehensiveness to the experts especially, in assisting the evaluation and certification of software products using the Common Criteria.**

***Keywords*— Put your keywords here, keywords are separated by comma.**

I. INTRODUCTION

The software industry is one of the fastest growing industries in the world due to the huge and currently increasing demand for software applications. The ways of software development can be by the standard, needs and company's circumstances. Software engineering is the application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software, and the study of these approaches; that is, the application of engineering to software (IEEE, 2004). Even though there are various ways of software development; the weaknesses from the security perspective in software development are always being criticized.

Common Criteria is one of the standards that emphasizes on the quality of security functionality in ICT products. It is an international standard for specification, implementation and evaluation of security in ICT products that is conducted in a rigorous manner. It can actually be described as a framework for consumers to specify the security requirement of the ICT product, also for the developers to claim the security functionality of their ICT product, for evaluators to evaluate and perform vulnerability assessment on the product to ascertain whether it meet the claims or not. The process in the software engineering and Common Criteria is leading to produce a good ICT product. Hence, the aim of this project is to develop a framework which has the combination of the Common Criteria and Software Engineering methodology to ensure that the developed software is complete with security assurance. Hence, this project is to develop a framework which are have the combination of the Common Criteria and Software Engineering methodology to make sure that the software that was developed complete with the security assurance.

II. BACKGROUND OF THE PROBLEM

In the Common Criteria evaluation and certification; various deliverables are required from software developers to be the evidences for evaluators and certifiers of Common Criteria. Software developers are usually familiar with software engineering deliverables but face difficulties in providing the deliverables that must be in line with the Common Criteria requirement. Software engineering deliverables lack the security requirements to be the evidences in the Common Criteria evaluation and certification. Hence, the aim of this project is to develop a framework which has the combination of the Common Criteria and Software Engineering methodology to ensure that the developed software is complete with security assurance.

III. PROJECT AIM

The aim of this research is to develop a framework between Common Criteria and software engineering's deliverables.

IV. SCOPE OF THE STUDY

1. The framework that has been developed only emphasizes the security practices
2. This research only focuses on the development of the software not the hardware.
3. The practices of software engineering only focus on the standard and industrial of software engineering.
4. The main focus only at the deliverables and process involved during software development because the software developers face difficulties to provide the deliverables or evidences that must in line with the

Common Criteria during Common Criteria evaluation and certification.

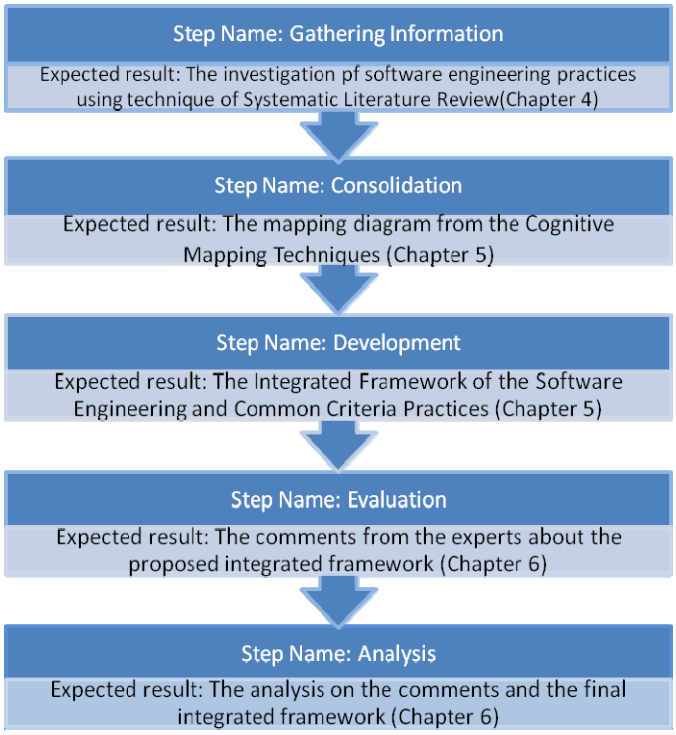
5. The focus only at the Technical and Implementation phase of the ISO/IEC 12207:2008 because both processes are related and contribute to the main technical aspects of software development.
6. The Composition (ACO) and Vulnerability Assessment (AVA) phase in the Common Criteria process is excluded from this research because the AVA deliverables are provided by the evaluators not the developers and the ACO phase are never occur yet and not relevant to focus this work unit in the research.

V. WHAT IS SOFTWARE ENGINEERING?

Software Engineering is profession dedicated to designing, implementing, and modifying software so that it is of higher quality, more affordable, maintainable, and faster to build. It is the application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software, and the study of these approaches; that is, the application of engineering to software (IEEE, 2004).

VI. METHODOLOGY

The methodology has five (5) steps to be performed which are Gathering Information, Consolidation, Formalization, Evaluation and Result.



VII. GATHERING INFORMATION

During the gathering information, the investigation of the software engineering will be conducted. Systematic reviews aim to present a fair evaluation of a research topic by using a trustworthy, rigorous, and auditable methodology. This chapter is using the technique of Systematic Literature Review that has been introduced by Barbara Kitchenham. The components of the Investigation of Software Engineering Practices are as in the Table I.

TABLE 1: COMPONENTS IN SYSTEMATIC LITERATURE REVIEWS (SOURCE: BARBARA KITCHENHAM, 2003)

No.	Components	Justification
1	Background	The rationale for the survey
2	The research questions	The research questions that the review is intended answer
3	The strategy	The research questions that the review is intended answer
4	Study Selection Criteria and Procedures.	Selection criteria determine criteria for including in, or excluding a study from, the systematic review. It is usually helpful to pilot the selection criteria on a subset of primary studies.
5	Study Quality Assessment Procedures.	The researchers should develop quality checklists to assess the individual studies. The purpose of the quality assessment will guide the development of checklists.
6	Data Extraction Strategy	This should define how the information required from each primary study would be obtained. If the data require manipulation or assumptions and inferences to be made, the protocol should specify an appropriate validation process.
7	Synthesis of the Extracted Data	This should define the synthesis strategy. This should clarify whether or not a formal meta-analysis is intended and if so what techniques will be used.

Herewith is the result of information after do the investigation on the standard and commonly practices industry of software engineering.

a) Standard of Software Engineering: ISO/IEC 12207:2008 System and software lifecycle processes

Besides meets those criteria such as the title of the studies, new initiatives (from a maximum of 10 years ago), focus on the standard, the process and deliverables from software engineering, written in English and the abstract is read and evaluated; the paper of Software & Systems Engineering Standards Committee of the IEEE Computer Society second edition 2008-02-01 on International Standard ISO/IEC 12207:2008; this International Standard ISO/IEC 12207:2008 establishes a common framework for software life cycle processes and be the references for the software industry. It contains processes, activities, and tasks that are to be applied during the acquisition of a software product or service and during the supply, development, operation, maintenance and disposal of software products. This International Standard applies to the acquisition of systems and software products and services, to the supply, development, operation, maintenance, and disposal of software products. It is also the latest standard of software engineering and software development and appropriate be the primary references in this project for the standard of software engineering that will be integrated with the Common Criteria.

b) Industrial of Software Engineering: Usage and Perceptions of Agile Software Development in an Industrial Context

This paper also meets the criteria as mentioned in the section 4.2.2. Besides that, the paper of Begel A and Nagappan N presents about the Agile Software

Development (ASD) methodology that had been used by the larger and well known industry; Microsoft and the research also proved that they covered a global perspective by send the surveys across North America, Asia and Europe. The industrial practices of software engineering that will be integrated with the Common Criteria methodology in this project is the ASD based justification on this paper.

VIII. CONSOLIDATION

The objective of consolidation process is to show the process deliverables from Common Criteria and software engineering practices are mapped before translated into a framework. The consolidation process has been conducted by mapping the deliverables from the Common Criteria and software engineering to see the similarities form both parties. The reason of excluding some process and deliverables also will be explained. The explanation will be using the mapping catalogue to show clearly the similarities of both deliverables and processes. This is how the Causal and Semantic mapping will be applied as they are described as using their own personal constructs to understand and interpret the events (Thomas H. Easter, 1999).

As mentioned in the Background Problem, the project is conducted because of the complexity from the developers' side to provide the deliverables to the Common Criteria. Therefore, the explanation is started by each phase in the Common Criteria. The explanation will use a mapping catalogue to determine the similarities of the processes and deliverables. The catalogue consists of four (4) areas which are the title, the Common Criteria process, the Software Engineering process and the mapping between both processes. Each area implemented to interpret the information about how both processes and deliverables are mapped together. This is how the concept of Causal and Semantic mapping are applied. The details of the mapping catalogue are described as the figure 2, figure 3 and figure 4.

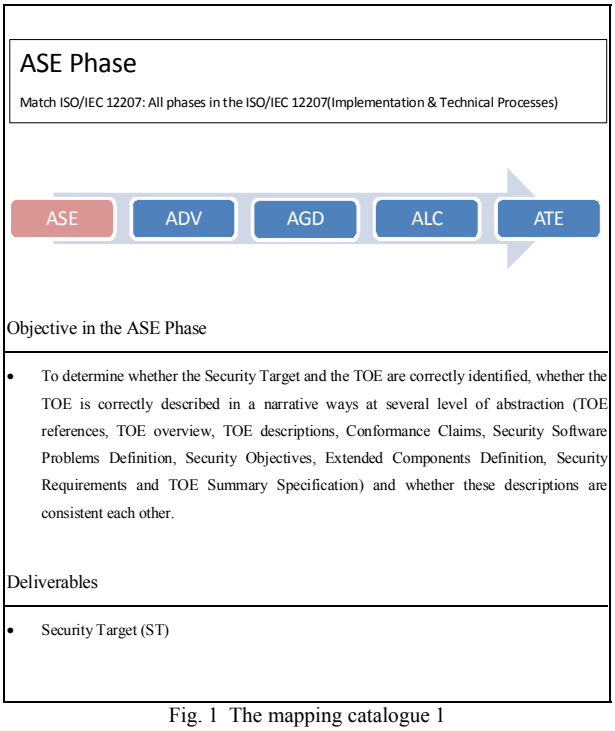


Fig. 1 The mapping catalogue 1

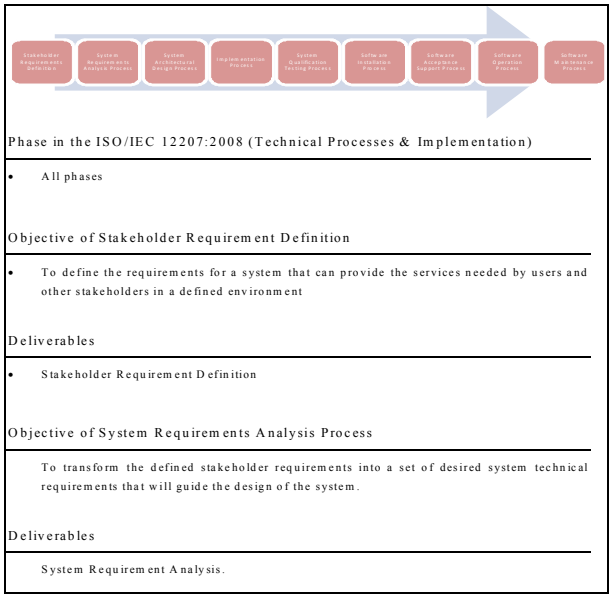


Fig. 2: The mapping catalogue II

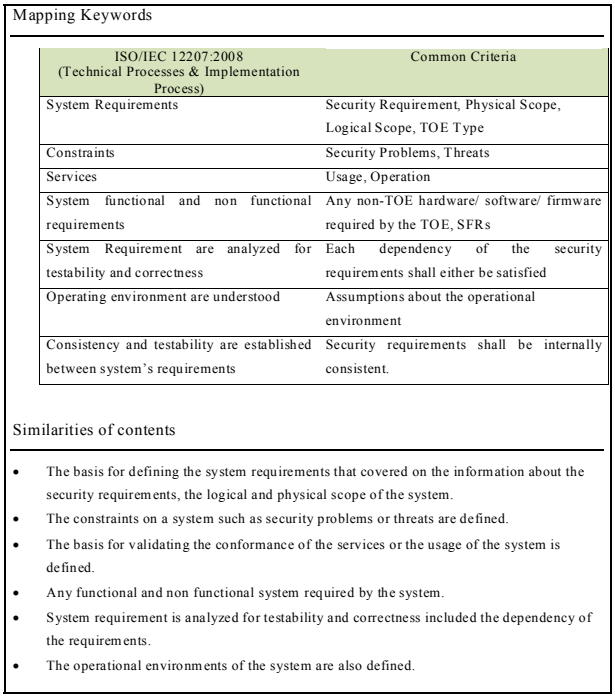


Fig. 3: The mapping catalogue III

IX. DEVELOPMENT FRAMEWORK

The development of framework to design the integrated framework framework is applying the Concept mapping. Concept mapping is useful in generating ideas, designing a complex structure, communicating complex ideas, aiding learning by explicitly integrating new and old knowledge, and assessing understanding or diagnosing misunderstanding (Dictionary, 2011).

The framework structure is using square, arrow and dashed lines as in the table II to show the connection and relationship in the framework. The usage of colors not only make the framework more attractive but guide the reader about the similar phases involved by looking the same color in the framework.

TABLE II: FRAMEWORK'S SYMBOL

Symbol	Meaning
	Phase, Title
	Direction
	Relationship

X. RESULT & DISCUSSION

a) The framework of Common Criteria and standard of software engineering (ISO/IEC 12207:2008)

The ISO/IEC 12207:2008 standard groups the activities that performed during the life cycle of a software system into seven process groups which are as below:

1. Agreement Processes
2. Organizational Project-Enabling Processes
3. Project Processes
4. Technical Processes
5. Software Implementation Processes
6. Software Support Processes
7. Software Reuse Processes

The process from the ISO/IEC 12207:2008 is only focus on the Technical and Software Implementation Processes. It is because of the two processes are the main process of the software development. Others such as the Agreement Processes, Organizational Project-Enabling Processes, Project Processes, Software Support Processes and Software Reuse Processes are not cover on the main technical software development. They are more as supportive processes to the Technical and Software Implementation Processes. The Agreement Processes, Organizational Project-Enabling Processes and Project Processes are more on the agreement and management of the organization's capability. The Software Reuse Processes also only focus on the maintenance of the domain, asset and management process to systematically exploit reuse opportunities and not in the main process of software development.

The Technical Processes contains eleven sub-processes which are as below:

1. Stakeholder Requirement Definition Process
2. System Requirements Analysis Process
3. System Architectural Design Process
4. Implementation Process
5. System Integration Process
6. System Qualification Testing Process
7. Software Installation Process
8. Software Acceptance Support Process
9. Software Operation Process
10. Software Maintenance Process
11. Software Disposal Process

System Integration Process and Software Disposal Process are not included in the project because their deliverables are not in the requirement of Common Criteria. Therefore, both of the processes are not in the scope of this project. In the sub-process of the Implementation Process under Technical Processes, it consists of Software Implementation Processes. This is the reason of including Software Implementation Process as the main focus because both processes (Technical Processes and Software Implementation Processes) are related and contribute to the main technical of software development. However, the processes under Software Implementation only involved with five (5) processes excluded the Software Integration Process and Software Qualification Testing Process. The reason of excluding the Software Qualification Testing Process under Implementation Process because of it already included under Technical Processes and it is not required to repeat it again under Implementation Process. Therefore, only nine (9) processes involved in the project which are:

1. Stakeholder Requirement Definition Process
2. System Requirement Analysis Process
3. System Architecture Design Process
4. Implementation Process
   i. Software Implementation Process
   ii. Software Requirement Analysis Process
   iii. Software Architectural Design Process
   iv. Software Detailed Design Process
   v. Software Construction Process
5. System Qualification Testing Process
6. Software Installation Process
7. Software Acceptance Support Process
8. Software Operation Process
9. Software Maintenance Process

The main structure of ISO/IEC 12207:2008 standard can be seen as below:

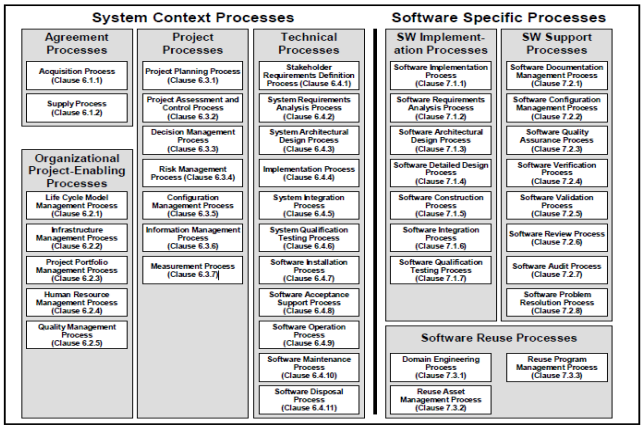


Fig. 4: The structure of ISO/IEC 12207:2008 (Source: IEEE Committee, 2008)

The framework of Common Criteria and ISO/IEC 12207:2008 as in the figure 5:

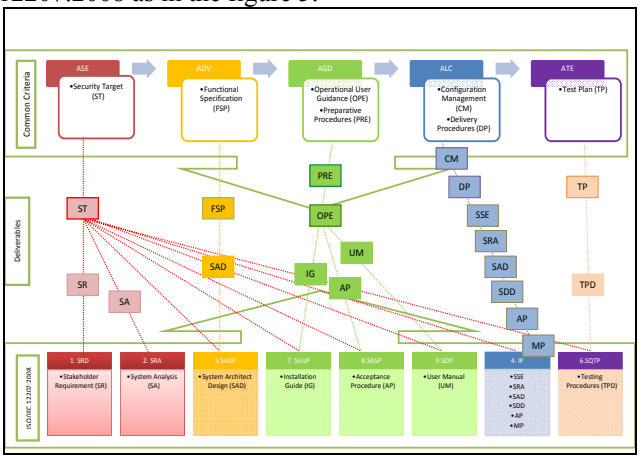


Fig 5: The framework of Common Criteria and ISO/IEC 12207:2008

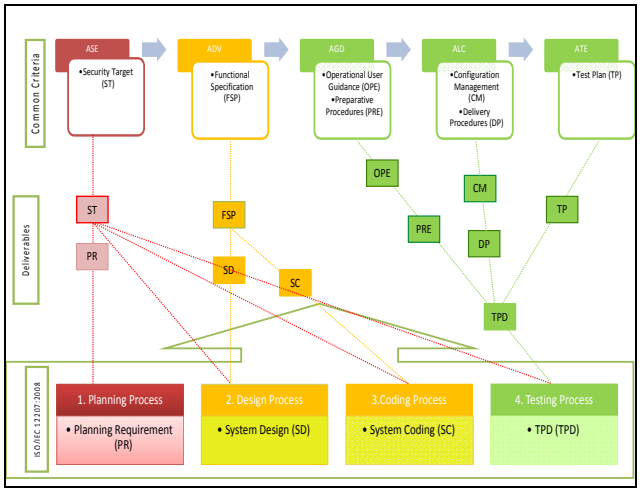
b) The framework of Common Criteria and commonly practices industry of software engineering (Agile Software Development)

The consolidation between process and deliverables between Common Criteria and Agile Software Development has being shown in the catalogue later. The Agile Software Development is found as the most acceptance software process in the industrial practices (Andrew Begel and Nachiappan Nagappan; 2007). The election of being this paper as the primary reference had been decided during the investigation on the software engineering practices in the Chapter 4.

The Agile Software Development consists of several processes which are started by planning, design, coding and test as in the table III. This project only emphasizes the process and deliverables taken in the Agile Software Development and the Scrum model that being used by Agile Software Development is not being concentrated.

TABLE III: THE PROCESS AND DELIVERABLES OF ASD

Agile Software Development	Deliverables
1. Planning Process	User stories value Acceptance Test Criteria Iteration Plan
2. Design Process	Simple Design CRC cards Spike Solutions prototypes
3. Coding Process	Pair Programming
4. Testing Process	Unit Test Continuous Integration Acceptance Testing



XI. EVALUATION & ANALYSIS

This section presents finding of evaluation that was conducted on the framework. The evaluation is using the questionnaires that were distributed to the respondents. The objective of the evaluation on the framework is to analyze the feasibility and understanding of the framework that was created.

a) Perceived ease of adoption among respondents

The figure 7 shows that the certifiers, evaluators and software developers are positive about the ease of adoption of the framework into the environment of Common Criteria evaluation and certification. The higher percentage goes to the certifiers with 88% followed by the software developers with 78% and the evaluators with 60%. Although of the percentage of evaluators are less than others, it still shows a positive percentage because it more than half percents from the sample of evaluators population. Therefore, it can be concluded that the framework is perceived ease of adoption in Common Criteria evaluation and certification among the sample population.

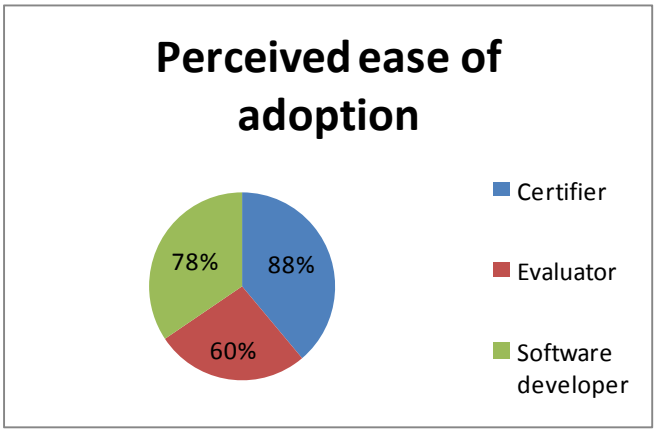


Fig 7: Perceived ease of adoption among respondents

b) Perceived utility from framework

The figure 8 shows the productivity level is among the highest and consistent from the respondents. The productivity of evaluation and certification of Common Criteria is perceived among the respondents from the framework. Most of the respondents also agree that less time required during the Common Criteria evaluation and certification when using the framework. The last top three utilities that could effect from the framework is the quality. Most of the respondents perceive that the quality of evaluation and certification of Common Criteria would be enhanced from the framework.

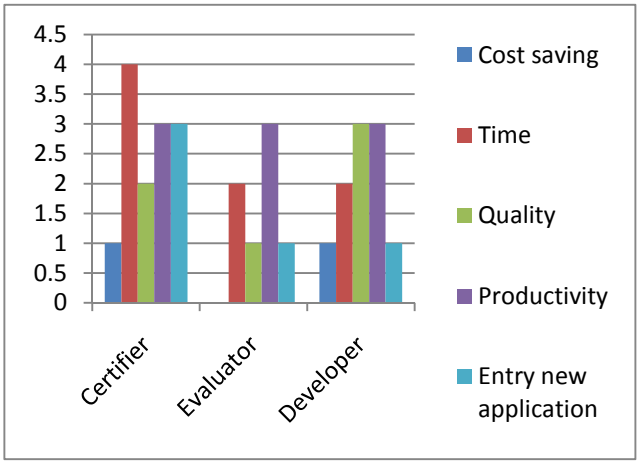


Fig. 8: Perceived utility from the framework

c) Perceived ease of adoption among respondents

The figure 9 shows that the certifiers, evaluators and software developers are positive about the ease of adoption of the framework into the environment of Common Criteria evaluation and certification. The higher percentage goes to the certifiers with 88% followed by the software developers with 78% and the evaluators with 60%. Although of the

percentage of evaluators are less than others, it still shows a positive percentage because it more than half percents from the sample of evaluators population. Therefore, it can be concluded that the framework is perceived ease of adoption in Common Criteria evaluation and certification among the sample population.

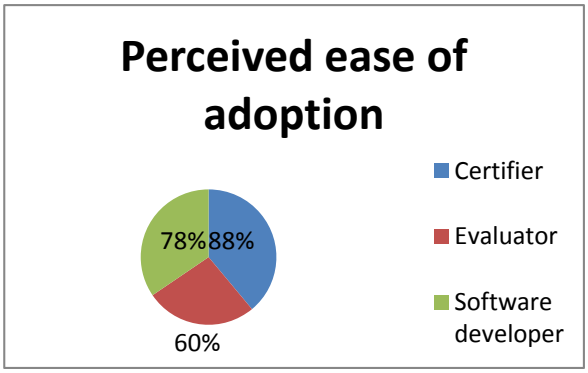


Fig. 9: Perceived ease of adoption among respondents

XII. CONCLUSION

The integrated framework will facilitate the software developers in producing software complete with security assurance because it is integration between software engineering and Common Criteria. When the integrated framework facilitate the software developers in developing a software complete with security assurance, it will increase the competency among software developers in developing more secure software and encourage more software product to be certified by Common Criteria. The integrated framework will simplify the process of evaluation and certification of security functionality of the product because the process and deliverables from software engineering will be mapped to the process and deliverables from Common Criteria. When the software developers aware about the existence of Common Criteria in the integrated framework, it will increase the awareness and confidence about the security software and Common Criteria certification among users.

ACKNOWLEDGMENT

Utmost gratitude and appreciation to those who have been directly or indirectly involved in the preparation of this project report. In particular, I would like to extend special thanks to my supervisor, Dr. Mohd Naz'ri Mahrin

REFERENCES

[1] Adams B, Schutter KD and Zaidman A (2009). The Journal of Systems and Software Using aspect orientation in legacy environments for reverse engineering using dynamic analysis. *The Journal of Systems & Software*. 82(4):668-684.

[2] Alonso F, Juristo N and Mat JL (1996). Software Engineering and Knowledge Towards a Common Life Cycle. *Structure* 1212:65-79.

[3] Antonellis P, Antoniou D and Kanellopoulos Y (2009). Code4Thought Project: Employing the ISO/IEC-9126 Standard for Software Engineering-Product Quality Assessment. *2009 13th European Conference on Software Maintenance and Reengineering*. 297-300.

[4] Aoyama M (1996). Beyond software factories: concurrent-development process and an evolution of software process technology in Japan. *Information and Software Technology*. 38(3):133-143.

[5] Ardi S and Shahmehri N (2009). Introducing Vulnerability Awareness to Common Criteria's Security Targets. *2009 Fourth International Conference on Software Engineering Advances*.419-424.

[6] Begel A and Nagappan N (2007). Usage and Perceptions of Agile Software Development in an Industrial Context: An Exploratory Study.

[7] First International Symposium on Empirical Software Engineering and Measurement (ESEM 2007). 255-264.

[8] Bimler D, Kirkland J and Pichler S (2004). Escher in color space: individual-differences multidimensional scaling of color dissimilarities collected with a gestalt formation task. *Behavior research methods, instruments, & computers : a journal of the Psychonomic Society, Inc*.36(1):69-76.

[9] Bond GW, Cheung E and Goguen HH (2005). Experience with Component-Based Development of a Telecommunication Service. *Development*. 298-305.

[10] Briand LC, El Emam K, Surmann D, Wieczorek I and Maxwell KD (1999). An assessment and comparison of common software cost estimation modeling techniques. *Proceedings of the 21st international conference on Software engineering - ICSE '99*. :313-322.

[11] Bru F-xavier, Frappin G, Legrand L and Merrer E. (2001) Building an Observatory of Course-of-Action in Software Engineering : Towards a Link between ISO / IEC Software Engineering Standards and a Reflective Practice. :185-200.

[12] Cheikhi L, Abran A, Suryn W, Superieure EDT and Ouest N-dame (2006). Harmonization of usability measurements in software engineering standards. *Cycle*. 3246-3251.

[13] De Win B, Scandariato R, Buyens K, Grégoire J and Joosen W (2009). On the secure software development process: CLASP, SDL and Touchpoints compared. *Information and Software Technology*. 1(7):1152-1171.

[14] Dong J, Wang J, Sun D and Lu H (2008). The Research of Software Product Line Engineering Process and Its Integrated Development Environment Model.

[15] Drehmer D (2001). A note on the evolution of software engineering practices. *Journal of Systems and Software*. 57(1):1-7

[16] Dyba T and Dingsoyr T (2008). Empirical studies of agile software development: A systematic review. *Information and Software Technology*. 50(9-10):833-859.

[17] Dyba T, Kampenes V and Sjoberg D (2006). A systematic review of statistical power in software engineering experiments. *Information and Software Technology*.48(8):745-755.

[18] Eia (1998) I. IEEE / EIA Standard International Standard Technology — Software life cycle processes. *Electronics*.

[19] Eia (1995) I. IEEE / EIA Guide Industry Implementation of International Standard ISO / IEC 12207 : 1995 (ISO / IEC 12207) Standard for Information Technology — Software life cycle processes — Life cycle data. *Source*.

[20] Emmerich EW. Chapter 2 Software Process Improvement Standards , Assessments and. *Engineering*.

[21] Engberts A, Ning JQ and Drive SW (1994). Transferring re-engineering technology to a software development and maintenance organization: an experience report. *Proceedings International Conference on Software Maintenance ICSM-94*. 100-108.

[22] Engelsma ES (2000) Incremental Systems Integration within Multidisciplinary Product Line Engineering Using Configuration Item Evolution Diagrams. *Mechatronics*.

[23] Finnie GR and Wittig GE (1997), Desharnais J-m. Estimating Software Development Effort with Case-Based Reasoning. *Analysis*.

[24] French VA and To J (1995) , eFacv Defence Svstem M-ce : d Pmcess Imgmvemea -rice RePr [llf. *Sciences-New York*.

[25] Goulao M and Brito e Abreu F (2007). Modeling the Experimental Software Engineering Process. *6th International Conference on the Quality of Information and Communications Technology (QUATIC 2007)*. 77-90.

[26] Habra N, Alexandre S, Desharnais J, Laporte C and Renault a (2008). Initiating software process improvement in very small enterprisesExperience with a light assessment tool. *Information and Software Technology*. 50(7-8):763-771.

[27] Han J and Zheng Y (2000). Security characterisation and integrity assurance for component-based software. *Proceedings International Conference on Software Methods and Tools. SMT*.61-66.

[28] Hwang S-myung (2004). A Design of Configuration Management Practices and CMPET in Common Criteria Based on Software Process. *Interface*.15504:481-490.

[29] Irish A, Vses S, Study C, Basri SB and Connor RVO (2010). Organizational Commitment Towards Software Process Improvement. *Assessment*.1456-1461.

[30] Keblawi F and Sullivan D. Applying the common criteria in systems engineering. *IEEE Security & Privacy Magazine*. 4(2):50-55

[31]Lee J, Lee J, Lee S and Choi B (2003). A CC-based security engineering process evaluation model. *Proceedings 27th Annual International Computer Software and Applications Conference. COMPAC 2003*. 130-135.

[32] Maletic JI, Howald a and Marcus a (2003). Incorporating PSP into a traditional software engineering course: an experience report. *Proceedings 14th Conference on Software Engineering Education and Training. "In search of a software engineering profession" (Cat. No.PR01059)*. 89-97.

[33] Management WS, Site W and Cycle L (2006). INTERNATIONAL STANDARD ISO / IEC Software Engineering — Recommended Engineering , Web Site Management , and. *Electronics*.

[34] Mark L and Rombach HD (1989). Generating customized software engineering information bases from software process and product specifications. *[1989] Proceedings of the Twenty-Second Annual Hawaii International Conference on System Sciences. Volume II: Software Track*. 587-595.

[35] Mason J (2003). Aligning workforce development & software process improvement strategy for accelerated adoption of software engineering capability. *Proceedings 16th Conference on Software Engineering Education and Training, 2003. (CSEE&T 2003).*70-77.

[36] Mellado D, Fernandezmedina E and Piattini M (2008). Towards security requirements management for software product lines: A security domain requirements engineering process. *Computer Standards & Interfaces*. 30(6):361-371.

[37] Mellado D, Fernandezmedina E and Piattini M (2007). A common criteria based security requirements engineering process for the development of secure information systems. *Computer Standards & Interfaces*. 29(2):244-253.

[38] Mellado D, Blanco C, Sánchez LE and Fernández-Medina E (2010). A systematic review of security requirements engineering. *Computer Standards & Interfaces*. 32(4):153-165.

[39]Mellado D, Fernandez-Medina E and Piattini M (2008). Security Requirements Variability for Software Product Lines. *2008 Third International Conference on Availability, Reliability and Security*. 1413-1420.

[40] Mellado D, Fernández-Medina E and Piattini M (2010). Security requirements engineering framework for software product lines. *Information and Software Technology*. 52(10):1094-1117.

[41] Montoni M, Santos G and Rocha AR (2006). Taba Workstation : Supporting Software Process Deployment Based on CMMI and MR-MPS . BR. 2006:249 - 262.

[42] Morali A, Zambon E, Houmb SH, Sallhammar K and Etalle S (2009). Extended eTVRA vs. security checklist: Experiences in a value-web. *2009 31st International Conference on Software Engineering - Companion Volume*. 130-140.

[43] Rabbi F, Wang H and MacCaull W (2010). YAWL2DVE: An Automated Translator for Workflow Verification. *2010 Fourth International Conference on Secure Software Integration and Reliability Improvement*. 53-59.

[44] Rao AS (2000). TECHNOLOGY ACCEPTANCE MODEL FOR COMPLEX TECHNOLOGIES IN A PERIOD OF RAPID CATCHING-UP.

[45] Scacchi W (1999). Experience with software process simulation and modeling. *Journal of Systems and Software*. 46(2-3):183-192.

[46] Seffah A, Donyaee M, Kline RB and Padda HK (2006). Usability measurement and metrics: A consolidated model. *Software Quality Journal*.14(2):159-178.

[47] Sherif K (2002). Domain engineering for developing software repositories: a case study. *Decision Support Systems*. 33(1):55-69.

[48] Siau K and Tan X (2005). Improving the quality of conceptual modeling using cognitive mapping techniques. *Data & Knowledge Engineering*. 55(3):343-365.

[49] Skandan S and Sidhardhan M (2009). Software Engineering Education at TCS Induction - An Experience Report. *2009 22nd Conference on Software Engineering Education and Training*. 16-19.

[50] Stallinger F, Dorling A, Henderson-sellers B, Box PO and Lefever B (2002). Software Process Improvement for Component-Based Software Engineering. *An Introduction to the OOSPICE Project The Practice of Component-based Soft- ware Engineering*. (Section 2).

[51] Standard I (2007). INTERNATIONAL STANDARD ISO / IEC. *October*. 2007;2007.

[52] Standard I (2008). INTERNATIONAL STANDARD ISO / IEC. 2008;2008.

[53] Tompkins F (1986). Integrating security activities into the software development life cycle and the software quality assurance process. *Computers & Security*.5(3):218-242.

[54] Trammell C (1996). The incremental development process in Cleanroom software engineering. *Decision Support Systems*. 17(1):55-71.

[55] Tripp L (1987). Taxonomy of software engineering standards: A development history. *Computer Standards & Interfaces*. 6(2):195-205.

[56] Tsai JJP and Liu A (2009). The Journal of Systems and Software Experience on knowledge-based software engineering : A logic-based requirements language and its industrial applications. *The Journal of Systems & Software*. 82(10):1578-1587.

[57] Upchurch RL and Sims-knight JE (2002). Session S2G PORTFOLIO USE IN SOFTWARE ENGINEERING EDUCATION : AN EXPERIENCE REPORT Session S2G. *Education*. 2-6.

[58] Ushakov I, Werling R and Nagano H (1993). Graphical representation of information in software engineering: ISO-IEC JTC1/SC7 framework, concepts and standards activity. *Proceedings 1993 Software Engineering Standards Symposium*. 209-218.

[59] Valett JD and Mcgarry FE. A S (1989). Software Measurement Experiences in the Software Engineering Laboratory. *Personnel*.148:137-148.

[60] Verner JM, Sampson J, Tosic V, Bakar N a A and Kitchenham B a (2009). Guidelines for industrially-based multiple case studies in software engineering. *2009 Third International Conference on Research Challenges in Information Science*. 313-324.

[61] Walker A (1998). Improving the quality of ISO 9001 audits in the field of software. *Information and Software Technology*. 40(14):865-869.

[62] Wang Y, King IG and Dorling A (1998). A Worldwide Survey of Base Process Activities Towards Software Engineering Process Excellence. *Society*. 1-4.

[63] Wangenheim C, Weber S, Hauck J and Trentin G (2006). Experiences on establishing software processes in small companies. *Information and Software Technology*. 48(9):890-900.

[64] Ware MS, Bowles JB and Eastman CM (2006). Using the Common Criteria to Elicit Security Requirements with Use Cases. *Proceedings of the IEEE SoutheastCon* 273-278.

[65] Zainol A and Mansoor S (2008). Investigation into Requirements Management Practices in the Malaysian Software Industry. *2008 International Conference on Computer Science and Software Engineering*. 292-295.