REVIEWS ON FUNCTIONAL SIZE MEASUREMENT IN MOBILE APPLICATION AND UML MODEL

Nur Atiqah Sia Abdullah[1], and Nur Ida Aniza Rusli[2]

[1]*Universiti Teknologi MARA(UiTM), Malaysia, atiqah@tmsk.uitm.edu.my*
[2]*Universiti Teknologi MARA(UiTM), Malaysia, idaaniza@gmail.com*

ABSTRACT. The increasing popularity of game industry has motivated some exploratory research in mobile games technology. This maturing technology, provided with complex functionality in development process requires software analyst to measure the size of mobile application properly, which relatively affect the software development cost and duration. Expert judgement is used in most of the mobile application development estimation. However, literatures suggest that formal effort estimation is more comprehensive and able to avoid misunderstandings. Therefore, most of literatures adapted Functional Size Measurement (FSM) in estimating the mobile development effort. Some literatures use FSM with UML modelling because UML model can represent the functional requirement of mobile application. This paper aims to review the FSM in mobile application and UML modelling in terms of measurement process and rules.

Keywords: functional size measurement, mobile application, UML modelling, software effort estimation

INTRODUCTION

Effort estimation is defined as the process of predicting the person-months required to develop software (Leung & Fan, 2002). It can be used further in estimating project duration and cost. Accurate estimation is important as statistics show that 45% of IT projects, including game application development, are over budget, 7% of the projects over schedule and 56% contained less value than predicted (Bloch et al., 2012). Game application is the most increasing in popularity, in which it is the largest category of time spent and has the largest number of application for Apple, Android and Windows Phone compared to education and entertainment (Khalaf, 2013). The increasing popularity of game industry has witnessed its development technology evolves from the game machine, video and PC games to mobile application (Herman et al., 2002). Most of the mobile application development is using expert judgement as the estimation technique (Lang et al., 2013). Expert judgement is carried out based on previous experience on similar projects. It causes difficulties for non-experts to understand how the estimation is conducted (Rush et al., 2001). Formal effort estimation is more comprehensive and able to avoid misunderstandings (Jørgensen & Grimstad, 2008).

Therefore, there are researches proposed to improve the estimation of mobile game application (Dusty, 2014; Tim, 2014). Due to the rapid development of mobile games and fast growing market, mobile game application demands more complex functionality and interactive 3D graphic design for customer satisfaction (Fritsch et al., 2006). Demands of high quality of 3D graphics will certainly imply extra cost for software game development companies

(Rapidsoft Technologies, 2015). Existing estimation metrics are unable to continue the estimation of effort based on the complex functionalities of 3D graphics (Abdullah et al., 2013; Lang et al., 2013; Giudice, 2013; Regnell et al., 2008). The existing methods do not consider the requirements for mobile application (de Souza & de Aquino Jr, 2014; Jošt et al., 2013; Nitze, 2013). Evidence in past literature also shows the modern game development requires a new set of software metrics to estimate the effort (Lang et al., 2013; Giudice, 2013; Abdullah et al., 2013). This paper reviews researches on FSM for mobile application and UML model.

RELATED WORKS

The following section reviewed FSM methods, which are adapted in estimating the software size based on mobile application properties and UML modelling.

Functional Size Measurement in Mobile Application

The section consists of research that maps the FPA rules in mobile application requirements to produce function point (de Souza & de Aquino Jr, 2014; Tunali, 2014; Nitze, 2013; Jost et al., 2013; Abdullah et al., 2013).

Due to emerging technology of mobiles such as smartphones and tablets, and lack of traditional metrics in adapting this mobile context, de Souza and de Aquino Jr (2014) attempt to introduce a mobile metric to estimate the effort of mobile application development. By proposing 23 mobile characteristics, these characteristics are grouped in productivity, performance, power, band, connectivity, context, graphic interface, and input interface for further estimation. FiSMA weighting factors (0.90 to 1.10) are assigned to each feature for the functional size processes. Reuse concept is included in the estimation but it does not discuss how the measurement is validated. Therefore, the reliability on applying the measurement in real world mobile application projects and reuse concept remains unknown.

Tunali (2014) provides a case study on mobile tablet application to be adapted in FPA measurement. Input measurement from code files are categorised into IFPUG base functional components (BFC). Mobile features such as activities on the application screen, activities involving pictures in gallery and audio files are categorised as the external inputs. The internal logical file and external logical files are captured from the databases such as picture and audio file. All files are then aggregated to IFPUG complexity (low, average, and high) to perform the function point counting. However, the mobile features are applicable only to small context of mobile application projects instead of complex application. Therefore, Tunali (2014) measurement might not applicable on the complex mobile game application.

Nitze, (2013) presents the idea on adapting the COSMIC Function Points in mobile application development by introducing 19 elements for the mobile development. The concept of the measurement was similar to the COSMIC FSM approach where the functional requirement, data groups, and also triggering events are identified and composed into the functional processes. With a small data set of mobile requirements, the authors defined standard "screen" elements such as home, setting and data input interface to obtain the functional process and data movements. Each element is aggregated to COSMIC read, write, entry or exit data movement for counting process. This method however can be improved by considering other factors such as networking, mobile toggling, and type of input devices despite of the "screen" element.

Jost et al. (2013) presented traditional software metrics for measuring mobile application programming code. The measurement structure begins with the development of small-scale application across Androids, iOS and Windows Phone platforms. From the visualisation of graphical elements for user interface, the authors introduced eight metrics to be applied in analysis of mobile application. The metrics include Depth of inheritance tree (DIT), Number

of children (NOC1), Lack of cohesion in methods (LCOM),Weighted methods per class (WMC), Number of methods (NOM), Number of classes (NOC), Lines of code (LOC) and Cyclomatic complexity (CC), which are used to evaluate the mobile application prototype. However, this method cannot be used in the early stage of development since it relies on the complete source code to get a proper measure. The authors also do not provide additional guidelines on how the proposed metric are applied to the mobile prototype and how they valued (high or low) the source code.

Abdullah et al. (2013) proposed the measurement of mobile application using the base component of COSMIC FSM. The measurement is specifically for game application. Angry Bird mobile game is used as a case study to measure mobile application. The UML modelling such as use case, behavioral model, component diagram, object diagram, and sequence diagram are used to include the functional requirements of mobile features including game scene, lighting, camera, sound, render, object and game model. COSMIC FSM data movement either entry, exit, read or write are identified in each UML modelling to be added later in COMIC formula for measuring the function point of mobile application.

Functional Size Measurement in UML Modelling

There are functional size measurements that adapted the UML modelling in FPA measurement based on the function point counting process (Irawati and Mustofa, 2012; Pow-Sang et al., 2009; Lavazza et al., 2008; del Bianco et al., 2008; Uemura et al., 1999).

Irawati and Mustofa (2012) proposed an approach to measure the software project from use case, class diagram, and relationship between use case and class diagrams. The complexity of measurement is based on IFPUG CPM 4.3.1 standard; the authors described the use cases as the representation of transactional function (TF) component and class diagram as data function (DF) component. Each non-repetitive attributes in TF and DF will correspond as data element type (DET) meanwhile number of classes diagram referred in each actor-to class process is mapped to File Type Reference (FTR) and Record Element Type (RET). Finally the total function point is counted by adding all function point for transactional function and data function. However, the proposed measurement focuses on the high level conceptual measurement, which does not include a clear measurement of class-to-class relationships.

Pow-Sang et al. (2009) proposed an estimation method by introducing rules to count the relationship between classes such as composition, association, aggregation and generalisation to be adapted in Function Point Counting Practice Manual 4.2.1. The identification of DET is a straightforward from the non-repetitive attributes from each class. The Logical File (LF) and Record Element Type (RET) are identified according to the relationships between classes. But the proposed rules only tested from the data developed by undergraduate students, which the validity of the method might not be accurate when it is applied in industry-based application.

del Bianco et al. (2008) and Lavazza et al. (2008) extended the function point using through the dimension of use case, component and sequence diagram. Use case diagram is used to make a distinctive interaction between main user; component diagram to support the transaction in the application and sequence diagram to weight each transaction in component diagram. The authors map the internal (ILF) and external (EIF) from the Logical Data and I/O component diagram. The attributes in class, the relationships between classes, and activities in sequence diagram are treated as DET, RET and FTR respectively. Lavazza et al. (2008) however provides a set of guidelines to build UML model based on FPA orientation. The main advantage is the measurement can be conducted without requiring any expertise since the authors mapped the rules to each creation of UML model in order to cater the entire information required by IFPUG principles.

Uemura et al. (1999) proposed function point measurement that considered class diagram and sequence diagram. This proposal calculated the elements of class diagram such as objects, classes and attributes to be transformed into sequence diagrams. Information will be extracted from class diagram and sequence diagram, which allow the counts of the DET from the class attributes, however the RET remains as one since RET cannot be counted from class and sequence diagram. Meanwhile the External Input (EI), External Output (EO) and External Inquiry (EQ) are counted based on the information extracted from sequence diagram. The weakness of this method is the restriction of RET value. RET will be remained to one, regardless of the complexity of objects, classes, or attributes in the class diagram or activities in the sequence diagram.

DISCUSSION

From the mobile application perspective, all reviewed methods use mobile components as the inputs for measurement. Tunali (2014), Nitze, (2013), Jost et al. (2013) and Abdullah et al. (2013) have presented the mobile components through case study to visualise the input elements but de Souza and de Aquino Jr (2014) generalised the characteristics of mobile application such as graphics, connectivity and performance as the base component. de Souza and de Aquino Jr's (2014) method allows the practitioners to conduct measurement in any types of requirements or mobile application genre. However, all reviewed methods might not be able to cater the software size estimation for mobile game application. It is due to the fact that mobile game application has its own distinctive challenges and complex development structure, which need different type of counting method.

With various mobile platforms such as iOS, Android and Windows Phone, the development of mobile game requires developers to have a set of components to be considered and specific tasks should be performed for the implementations. Since game development involves complex and long codes of program, game engine is highly useful to simplify and organise the game design and to perform important components such as object models, animation, and three-dimensional games. Game engine is a framework designed for creation and development of game which contained with advanced functionalities will be a good representation of mobile game development requirement. However, none of the reviewed methods considered the game engine components as the counting elements.

The basic idea of software measurement is presenting the elementary structure of software requirements as the inputs to the estimation. UML modelling can provide both static and dynamic properties in the input data by including the elements such as classes, components and the relationships with message as the key factors to construct measurement. Designing a mobile game application can be a complex process, which involves numerous requirements. Furthermore, game development is less structured compared to other software (Taylor et al., 2007). The utilisation of UML concept as a game framework managed to support game components and formalised all activities such as game modelling positions, animation and game environment.

The increasing use of UML model as the abstraction of software projects, which independent to any programming language and able to capture functionalities at early stage of development, has led to the researchers to adapt the UML design in function point counting process. All reviewed literatures consider class diagram as input to the measurement. Only Pow-Sang et al. (2009) and Lavazza et al. (2008) described the details of the relationship in class diagram such as composition, association, aggregation and generalisation. The structure of class relationships should be defined and included in weighting complexity since each relationship can formulate different kind of requirements and these factors must be considered in counting process in order to obtain more accurate in measurement method.

Finally, comparing all UML modelling methods, only Lavazza et al. (2008) illustrates a guideline to build the UML models in function point context. This method exploits the information in use case diagram to be transformed into class diagram, component diagram and sequence diagram. The designed rules for UML creation is the key factor to cater complete information about requirements that can be mapped in FPA counting from the UML model.

CONCLUSION

This paper reviews on existing methods that use the UML modelling and mobile requirement to expand the function point counting. Software metric characteristics from both categories are used as the guideline for future proposal. The next stage of research will involve the proposal of new metric for mobile game. It will provide with a procedure to construct a UML modelling based on the specification described on Petridis's methodology for game engine. Class and component diagrams will represent the internal and external boundaries that interact the application and rules to map the UML element to IFPUG are defined in order the process to measure the functional requirements are properly catered. The mapping will determine element for counting the function point such as data function, transactional function and use IFPUG complexity for sizing process. Lastly, the validation will be conducted by group of expertise in mobile game development to observe whether the new metric able to size the function point from different requirements.

ACKNOWLEDGMENTS

The authors express appreciation to Jabatan Pembangunan Sumber Manusia (JPbSM), Universiti Teknologi MARA for sponsoring this paper.

REFERENCES

Abdullah, N.A.S., Rusli, N.I.A., & Ibrahim, M.F. (2013). A Case Study in COSMIC Functional Size Measurement: Angry Birds Mobile Application. *Proceedings of the IEEE Conference on Open Systems*, 139-144. doi: 10.1109/ICOS.2013.6735063.

de Souza, L. S. & de Aquino Jr, G.S. (2014). Estimating the Effort of Mobile Application Development. *Proceedings of Second International Conference on Computational Science and Engineering*, 45-63. doi:10.5121/csit.2014.4405.

del Bianco, V., Gentile, C., & Lavazza, L. (2008). An evaluation of function point counting based on measurement-oriented models. *Proceedings of 12th International Conference on Evaluation and Assessment in Software Engineering.* Retrieved from http://ewic.bcs.org/content/ConWebDoc/19532

Dusty (2014). Beginner's guide to estimating the budget for your first 'big' indie game. Retrieve from https://www.gameacademy.com/beginners-guide-estimating-budget-first-big-indie-game/

Fritsch, T., Ritter, H., & Schiller, J. (2006). Mobile phone gaming (a follow-up survey of the mobile phone gaming sector and its users). In R. Harper, M. Rauterberg & M. Cambetto (Eds.), *Lecture Notes in Computer Science. Entertainment Computing – ICEC 2006* (pp.292-297). Berlin, Germany: Springer-Heidelberg. doi: 10.1007/11872320_35.

Giudice, D.L. (2013). Forrester: modern app development requires a modern set of metrics. Retrieved from http://www.computerweekly.com/opinion/Forrester-Modern-application-development-requires-a-modern-set-of-metrics

Irawati, A. R., & Mustofa, K. (2012). Measuring Software Functionality Using Function Point Method Based On Design Documentation. *International Journal of Computer Science Issues*, 9(3), 124-130. Retrieved from http://ijcsi.org/papers/IJCSI-9-3-1-124-130.pdf

Jørgensen, M., & Grimstad, S. (2008). How to avoid impact from irrelevant and misleading information when estimating software development effort. Retrieved from http://www.myai.org/blog/wp-content/uploads/2012/12/Estimation-Irrelevant-and-Misleading-Information.pdf

Jošt, G., Huber, J., & Heričko, M. (2013). Using Object Oriented Software Metrics for Mobile Application Development. *Second Workshop on Software Quality Analysis, Monitoring, Improvement and Applications*.

Khalaf S. (2013). Flurry five-year report: It's an app world. The web just lives in it. Retrieved from http://blog.flurry.com/?Tag=Usage%20Statistics.

Lang, M., Conboy, K., & Keaveney, S. (2013). Cost estimation in agile software development projects. In R. Pooley, J. Coady, C. Schneider, H. Linger, C. Barry & M. Lang (Eds.), *Lecture Notes in Information System Development: Reflections, Challenges and New Directions* (pp.689-706). New York, USA: Springer-Verlag. doi: 10.1007/978-1-4614-4951-5_55.

Lavazza, L. A., Del Bianco, V., & Garavaglia, C. (2008). Model-based functional size measurement. *Proceedings of the Second ACM-IEEE International Symposium on Empirical Software Engineering and Measurement*, 100-109. doi:10.1145/1414004.1414021.

Leung, H., & Fan, Z. (2002). Software cost estimation. *Handbook of Software Engineering*, Hong Kong Polytechnic University.

Nitze, A. (2013). Measuring mobile application size using COSMIC FP. *Proceedings of DASMA Metrik Kongress Conference*. Retrieved from http://www.dasma.org/contray/html/mitgliedsbereich/metrikon-downloads/metrikon-2013.html

Pow-Sang, J. A., Gasco, L., & Nakasone, A. (2009). A Function Point Logic File Identification Technique Using UML Analysis Class Diagrams. In D. Ślęzak, T.H. Kim, A. Kiumi, T. Jiang, J. Verner, & S. Abrahão (Eds.), *Lecture Notes in Communications in Computer and Information Science: Vol.* 59. *Advances in Software Engineering* (pp.160-167). Berlin, Germany: Springer-Verlag. Retrieved from http://link.springer.com/chapter/10.1007%2F978-3-642-10619-4_20

Rapidsoft Technologies (2015). Estimating the cost of enterprise mobile application development. Retrieve from http://www.rapidsofttechnologies.com/blog/index.php/estimating-cost-enterprise-mobile-application-development/

Regnell, B., Svensson, R. B., & Wnuk, K. (2008). Can we beat the complexity of very large-scale requirements engineering? In B. Paech & C. Rolland (Eds.), *Lecture Notes in Computer Science: Vol.* 5025. Requirements *Engineering: Foundation for Software Quality* (pp.123-128). Berlin, Germany: Springer-Verlag. doi: 10.1007/978-3-540-69062-7_11.

Rush, C., & Roy, R. (2001). Expert judgement in cost estimating: Modelling the reasoning process. *Journal of Concurrent Engineering*, 9(4), 271-284. doi: 10.1177/1063293X0100900404.

Taylor, M. J., Baskett, M., Hughes, G. D., & Wade, S. J. (2007). Using soft systems methodology for computer game design. Systems Research and Behavioral Science, 24(3), 359-368. doi: 10.1002/sres.805

Tim (2014). How much do mobile games cost? A rough price guide to having your iOS or Android game developed. Retrieve from http://teamcooper.co.uk/blog/how-much-does-a-mobile-game-cost-to-develop/

Tunali, V. (2014). Software Size Estimation Using Function Point Analysis-A Case Study for a Mobile Application. Symposium of Mühendislik ve Technology, Turkey. Retrieved from http://www.academia.edu/7912949/Software_Size_Estimation_Using_Function_Point_Analysis_A_Case_Study_for_a_Mobile_Application

Uemura, T., Kusumoto, S., & Inoue, K. (1999). Function point measurement tool for UML design specification. *Proceedings of Sixth International Software Metrics Symposium*, 62-69. doi: 10.1109/METRIC.1999.809727.