2016 International Conference on Platform Technology and Service (PlatCon)

Proceedings

15-17 February 2016
Jeju, Korea

IEEE Catalog Number: CFP16F03-ART (Xplore)
ISBN: 978-1-4673-8685-2 (Xplore)

IEEE Catalog Number: CFP16F03-CDR (CD)
ISBN: 978-1-4673-8684-5 (CD)

Improving Use Case Point(UCP) based on Function Point(FP) Mechanism

Bo Kyung Park
SELab., Dept. of Computer and Information Communications, Hongik University
2639, Sejong Campus, 30016, Korea
park@selab.hongik.ac.kr

So Young Moon
SELab., Dept. of Computer and Information Communications, Hongik University
2639, Sejong Campus, 30016, Korea
msy@selab.hongik.ac.kr

R. Young Chul Kim
Dept. of Computer and Information Communications, Hongik University
2639, Sejong Campus, 30016, Korea
bob@hongik.ac.kr

***Abstract*—The cost of error correction has been increasing exponentially with the advancement of software industry. To minimize software errors, it is necessary to extract accurate requirements in the early stage of software development. In the previous study, we extracted the priorities of requirements based on the Use Case Point (UCP), which however revealed the issues inherent to the existing UCP as follows. (i) The UCP failed to specify the structure of use cases or the method of write the use cases, and (ii) the number of transactions determined the use case weight in the UCP. Yet, efforts taken for implementation depend on the types and number of operations performed in each transaction. To address these issues, the present paper proposes an improved UCP and applies it to the prioritization. The proposed method enables more accurate measurement than the existing UCP-based prioritization.**

Keywords—Function Point; Improved UCP; Use Case Point; Use Case Priority;

I. INTRODUCTION

The damage associated with software defects has increased with the advancement of software industry. Figure 1 shows the cost of error correction in each step of software development.

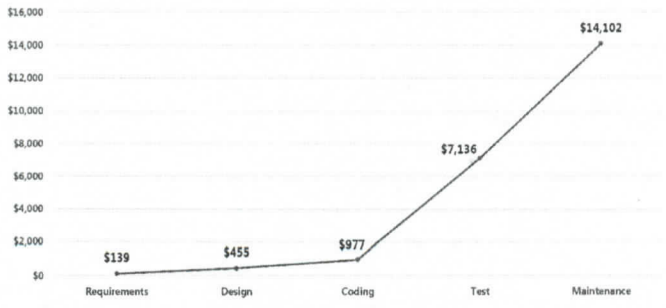

Fig. 1. Cost of error correction for each stage of software development[1]

The cost of error correction exponentially increases in line with the detection stages. Specifically, the cost of error correction in the maintenance stage is 100-fold higher than that in the requirement stage. To minimize software errors, accurate requirements should be extracted in the early stage of software development. Figure 1 shows the cost of error correction in each stage of software development.

In the requirement stage, the prioritization of requirements is important in that high quality products should be developed within tight resource and time constraints [2]. Prioritization of requirements facilitates the development of high-quality software whilst minimizing the cost. Previous studies extracted and verified the priorities of requirements based on the Use Case Point (UCP) [2, 3]. That is, a UCP was extracted for each use case. Then, the extracted UCP was used to prioritize the use case. Yet, the foregoing method has the issues inherent to the existing UCP as follows. First, the UCP failed to specify the structure of the use case or the method of writing the case. Thus, the use case models and specifications could vary. Second, the use case weight was determined based on the number of transactions in the UCP. However, the efforts taken for implementation in each transaction were dependent on the types and number of operations performed [4]. In effort estimation, the use case is reflected in a same size in line with the scopes, and thus measured inaccurately.

To address the aforementioned issues, the present study proposes an improved UCP and applies it to the prioritization. This paper comprises of the following chapters. Chapter 2 covers related works on UCP-based prioritization method. Chapter 3 describes the method of calculating priorities based on the improved UCP. Chapter 4 compares the existing method with the proposed one. Finally, chapter 5 presents the conclusion and suggestion for a further study.

II. RELATED WORKS

For the prioritization of use cases, Karner's use case point estimation is used [2, 3]. The size of software is measured quantitatively using the actors and use cases in the use case diagram. Unlike the existing UCP, the point of each use case is estimated. That is, the sum of actor and use case weights is not calculated. Figure 2 shows the UCP-based prioritization.

978-1-4673-8685-2/16/$31.00 ©2016 IEEE

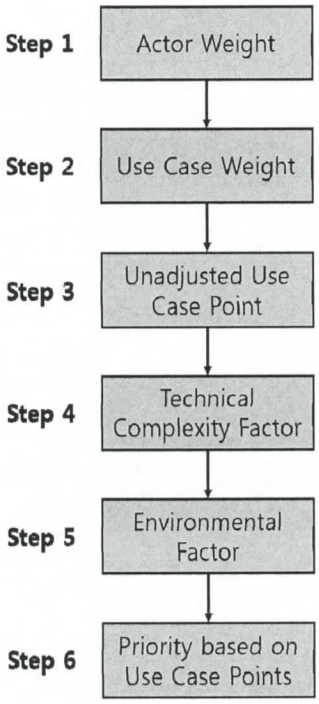

Fig. 2. Priority based on Use Case Points

The first step in the prioritization of use cases is to estimate the actor weight. Based on the use case points, the actor weights are estimated to be Simple(1), Average(2), and Complex(3). The actor weight is calculated in each use case for the prioritization of requirements. The second step is to estimate the use case weight. The use case weight is calculated based on the number of transactions of the use case. When the number of transactions is no more than 3, the use case weight is Simple. 4 ~ 7 transactions become the Average use case weight. 8 or more transactions correspond to the Complex use case weight. All use case weights add up to the final use case weight. By contrast, in the prioritization based on use case points, each use case weight is used to extract the priorities. In the third step, the actor weight and the use case weight add up to the unadjusted use case points. In the fourth step, the technology complexity factor determines a weight somewhere between 0(no effects) and 5(large effects) from the aspect of the entire system. In the fifth step, the environment factor applies a weight somewhere between 0 and 5 based on the UCP classification. Upon completion of calculation up to the fifth step, the extracted use case points are compared for the UCP-based prioritization.

III. PRIORITIZATION BASED ON IMPROVED UCP

The present study improves the existing UCP to prioritize the requirements. This chapter defines the improved UCP method, and describes the method of prioritization. This paper applies the prioritization method based on the improved UCP to a vehicle supplies management system. Figure 3 shows a use case diagram of the vehicle supplies management system.

Fig. 3. Use Case Diagram for the Vehicle Supplies Management System

The use case point (UCP) was suggested by Karner [5]. The UCP is derived from the use case modeling. Here, the extent to which the derived UCP reflects the accurate scale of software is crucial. Yet, all the existing UCPs have the following issues. (i) The UCP does not specify the method of writing the use case. Therefore, the use case model and specification may vary with analysts, leading to different sizes derived from a use case. (ii) The UCP determines the use case weight based on the number of transactions. Still, in each transaction, the types and number of operations performed determine the efforts taken for implementation [4]. However, these use cases are reflected in a same size in line with the scopes, and thus inaccurately measured. (iii) The UCP failed to consider the Include and Extends relations of use cases. To address these issues, the types and weight of actors and use cases are classified in this paper. Also, a weight of 0.25 is added to a use case which involves any Include or Extend relation. This method was suggested by Periyasamy et al. [6].

Step 1: Actor Weight

Based on the UCP, actor weights are classified into three types: Simple (1), Average (2) and Complex (3). Yet, Periyasamy's method classified the actors [6]. Periyasamy's actor weight is improved in this paper. In [6], the weight was allocated based on the types of actors and the number of transactions performed by the actors. Still, there are 7 types of actors (i.e. Very simple, Simple, Less average, Average, Complex, Very complex and Most complex). Moreover, it is necessary to analyze primary and secondary actors. This method is time consuming and complex because of the extraction of use case narrative factors. Hence, this paper adopts 5 types of improved actors. Table 1 shows the actor weight classification.

TABLE I. ACTOR WEIGHT CLASSIFICATION

Actor Type	*Classification of Actors*	*Weight*
Very Simple	Specialized Actor	0.5
Simple	Actor with 1<number of associations<=3	1
Average	Actor with 3<number of associations<=5	1.5
Complex	Actor with 5<number of associations<=8	2
Very Complext	Actor with number of associations>8	2.5

Step 2: Use Case Weight

Table 2 shows the classification of use case weight.

TABLE II. USE CASE WEIGHT CLASSIFICATION

Use Case Type	*Classification of Actors*	*Weight*
Simple	Number of transactions <= 2	0.5
Average	2 < Number of transactions <= 4	1
Complex	4 < Number of transactions <= 6	2
Very Complex	Number of transactions > 6	3

As in the actor weight, a different weight is allocated to each use case. In [6], the use case weight was defined as a number directly associated between an actor and a use case. This paper applies the use case type as it is, whereas the criteria for the classification of use case is refined. Also, a weight of 0.25 is added to a use case that involves any Include or Extend relation. Each extracted use case is prioritized based on its weight.

Step 3: Unadjusted Use Case Point (UUCP)

The unadjusted use case point is the sum of actor and use case weights. Table 3 shows the values of Unadjusted Actor Weight(UAW) and Unadjusted Use Case Weight(UUCW).

Step 4: Technical Complexity Factor (TCF)

TCF comprises 13 items in total. In TCF, each factor is given a weight somewhere between 0 and 5. 0 indicates no effects, whereas 5 represents large effects. In this paper, the weight is given by the 0.5 for the accurate prioritization of requirements.

Step 5: Environmental Factor (EF)

The environment factor comprises 8 items, with the weight (0 ~ 5) applied based on the classification. This paper does not consider the environment factor weight as all the use cases are given a same value (3).

Step 6: Use Case Priority

Following the completion of all estimation, each use case point is calculated. UCP is the product of UUCP, TCF and EF. The use cases are prioritized by comparing the extracted UCP values. Table 4 shows the final priorities.

TABLE III. UNADJUSTED USE CASE POINT

		Unadjusted Actor Weight(UAW)		Unadjusted UseCase Weight(UUCW)						UUCP
No	Use Case	(Manager) Actor Weight	Actor Weight	Basic Flow	Alternative Flow	Exceptional Flow	Include/ Extends	Total Transaction	Use Case Weight	
UC1	Login	1	1	1	1	1	0	3	1	2
UC2	Customer Register	1	1	1	1	0	0	2	0.5	1.5
UC3	Customer Update	1	1	1	0	0	0	1	0.5	1.5
UC4	Customer Retrieve	1	1	1	1	0	0.25	2.25	1	2
UC5	Customer Delete	1	1	1	0	0	0	1	0.5	1.5
UC6	Stock Register	3	1	1	0	1	0	2	0.5	1.5
UC7	Stock Retrieve	3	1	1	1	0	0.25	2.25	1	2
UC8	Stock Delete	3	1	1	0	0	0	1	0.5	1.5
UC9	Sale Register	3	1	1	0	0	0	1	0.5	1.5
UC10	Sale Retrieve	3	1	1	1	0	0.25	2.25	1	2
UC11	Sale Update	3	1	1	1	0	0	2	0.5	1.5
UC12	Sale Delete	0.5	0.5	1	1	0	0	2	0.5	1
UC13	Product Register	3	1	1	0	0	0	1	0.5	1.5
UC14	Product Retrieve	3	1	1	1	0	0.25	2.25	1	2
UC15	Product Delete	3	1	1	0	0	0	1	0.5	1.5
UC16	Inventory Retrieve	1.5	1.5	1	1	0	0.25	2.25	1	2.5
UC17	Income Retrieve	1.5	1.5	1	3	0	0.25	4.25	2	3.5
UC18	Expense Register	1	1	1	0	1	0	2	0.5	1.5
UC19	Expense Update	1	1	1	0	0	0	1	0.5	1.5
UC20	Expense Retrieve	1	1	1	3	0	0.25	4.25	2	3
UC21	Rexpense Delete	0.5	0.5	1	0	0	0	1	0.5	1
UC22	Print	0	0	1	1	0	1.5	3.5	2	2

TABLE IV. THE FINAL RESULT

		TCF1	TCF2	TCF3	TCF4	TCF5	TCF7	TCF9	TCF11	TCF13	TCF Value	UCP	Priority
		0~5											
No	Use Case	2	1	1	1	1	0.5	1	1	1			
UC1	Login	0	2	1.5	0	0	1.5	0	2	1	8	16	12
UC2	Customer Register	0	2	1.5	1	2	2	0	0	0	8.5	12.75	14
UC3	Customer Update	0	1	2	1	2	2	3	0	0	11	16.5	11
UC4	Customer Retrieve	0	1	2	0	0	2	1	0	0	6	12	15
UC5	Customer Delete	0	1	1.8	0	0	0	0	0	0	2.8	4.2	22
UC6	Stock Register	0	1	1	0	0	1	0	0	0	3	4.5	21
UC7	Stock Retrieve	0	3	3	4	1	4	1	2	0	18	36	1
UC8	Stock Delete	0	2	3	2.5	1	3	1	0	0	12.5	18.75	8
UC9	Sale Register	0	1	2	0	0	1	0	0	0	4	6	20
UC10	Sale Retrieve	0	1	1	0	1	2	0.5	0	0	5.5	11	16
UC11	Sale Update	0	3	3	4	1	4	1	2	1	19	28.5	4
UC12	Sale Delete	0	3	3	1	1	4	3	1.5	1	17.5	17.5	10
UC13	Product Register	0	2	3	0	1	2	1	2	1	12	18	9
UC14	Product Retrieve	0	2	3	1	0	3	1	2	1	13	26	5
UC15	Product Delete	0	2.5	3	2.5	1	3	1	0	0	13	19.5	7
UC16	Inventory Retrieve	0	2	3	2	1	3	1	0	0	12	30	3
UC17	Income Retrieve	0	1	2	0	0	0	0	0	0	3	10.5	18
UC18	Expense Register	0	1	2.3	0	0	1	2	0	0	6.3	9.45	19
UC19	Expense Update	0	2	3	3	1	4	1	0	0	14	21	6
UC20	Expense Retrieve	0	1	3	1	1	3	2	0	0	11	33	2
UC21	Rexpense Delete	0	2	3	0	1	2	3	0	0	11	11	16
UC22	Print	0	1	2	0	1	1.5	1	0	0	6.5	13	13

IV. PRIORITIZATION BASED OD EXISTING UCP VS. IMPROVED UCP

This chapter compares the priorities based on the UCP extracted in [2] with those based on the improved UCP. Table 5 shows the results of this comparative analysis. The analysis of priorities highlights the following: (i) A difference by 10 or more between the two methods is found in 6 items, i.e. Stock Delete, Sale Retrieve, Product Register, Product Delete, Income Retrieve and Expense Register. The improved UCP weights are defined to be 0.5~2.5 for actors and 0.5~3 for use cases. The existing UCP weights are defined to be 1~3 for actors and 5, 10 and 15 for use cases. As different weights are applied, the UCP values extracted are different between the two methods. (ii) The actor weights based on the improved UCP are measured using the number of direct associations between actors and use cases. The use case weight varies with the number of transactions. Thus, the priority based on the improved UCP differs from that of the existing UCP. (iii) The improved UCP considers Include and Extend relations in extracting the UCP. The improved UCP is a revised version of the existing UCP method. Also, the actor and use case weights are classified further in comparison to the existing UCP. Hence, the improved UCP is likely to extract more accurate UC points than the existing UCP. Further studies need to verify which one is accurately measured based on a real implementation.

TABLE V. RESULTS OF COMPARISON

No	Use Case	The Priority based on UCP	The Priority based on Improved UCP
UC01	Login	14	12
UC02	Customer Register	16	14
UC03	Customer Update	8	11
UC04	Customer Retrieve	13	15
UC05	Customer Delete	19	22
UC06	Stock Register	15	21
UC07	Stock Retrieve	7	1
UC08	Stock Delete	18	8
UC09	Sale Register	15	20
UC10	Sale Retrieve	4	16
UC11	Sale Update	9	4
UC12	Sale Delete	12	10
UC13	Product Register	16	9
UC14	Product Retrieve	6	5
UC15	Product Delete	17	7
UC16	Inventory Retrieve	4	3
UC17	Income Retrieve	2	18
UC18	Expense Register	10	19
UC19	Expense Update	3	6
UC20	Expense Retrieve	1	2
UC21	Rexpense Delete	11	16
UC22	Print	20	13

V. CONCLUSION

This paper improves the existing UCP-based prioritization. The improved use case points are used here to prioritize the use cases. The improved UCP method measures the actor weights based on the number of direct associations between actors and use cases. The use case weights are measured based on the number of transactions. Notably, the number of transactions is adjusted, or lowered for the use case weight compared to the existing UCP. As a result, 6 items show a significant difference. A further study needs to implement a vehicle supplies management system and thus to verify the accuracy of measurement.

ACKNOWLEDGMENT *(Heading 5)*

This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (NRF-2013R1A1A2011601) and the Human Resource Training Program for Regional Innovation and Creativity through the Ministry of Education and National Research Foundation of Korea (NRF-2015H1C1A1035548).

REFERENCES

[1] Boehm, Barry, and Victor R. Basili. "Software defect reduction top 10 list." Foundations of empirical software engineering: the legacy of Victor R. Basili 426 (2005).

[2] So Young Moon, Bo Kyung Park, and R. Young Chul Kim, "Verification of Requirements Extraction and Prioritization using Use Case Points", ITCS(Information Technology and Computer Science), ASTL13, pp. 100-104, July 2012.

[3] Bokyung Park, Soyoung Moon, Dongho Kim, Chaeyeon Seo, R. Youngchul Kim, "A Study on Extraction of Goal Oriented Use Case Based Requirements", Proceedings of 2012 Korea Conference on Software Engineering(2012).

[4] SunKyung Lee, DongWon Kang, Doo-Hwan Bae, "Software Effort Estimation based on Use Case Transaction", Journal of KIISE, Vol. 41, No. 11, November 2014.

[5] Karner, G, "Resource Estimation for Objectory Projects", Objective System SF AB(Copyright owned by Rational Software), 1993.

[6] Periyasamy, Kasi, and Aditi Ghode. "Cost estimation using extended use case point (e-UCP) model." Computational Intelligence and Software Engineering, 2009. CiSE 2009. International Conference on. IEEE, 2009.