Measuring Software Functionality Using Function Point Method Based On Design Documentation

Anie Rose Irawati [1] and Khabib Mustofa [2]

[1] Department of Computer Science, University of Lampung
Bandar Lampung, 35145, Indonesia

[2] Departement of Computer Science and Electronics, Gadjah Mada University
Yogyakarta, 55281, Indonesia

Abstract

Estimated value of software as agreed by the end user and the developer team should be expressed in a certain magnitude, one of which is the measure of functionality. Function Point (FP) Method is one of the methods used to obtain the size of the functionality and can be used to estimate cost, duration, and amount of resources required by a software project. However, Function Point measurement is not simple and requires expertise in software analysis. Furthermore, the results of the calculation are considered valid if it is verified by someone with International Function Point User Group (IFPUG) certification. This research aims at designing and implementing a system that makes users convenient in analyzing software functionality size based on FP method referring to IFPUG CPM 4.3.1 standards. The system helps users to perform FP analysis in a faster and easier way without sacrificing accuracy. The input for the system is XMI document resulting from software design documentation derived from UML documents. The study also reveals that the more complete UML documents of the software in the project, the more accurate the FP calculation results obtained.

Keywords: *Software measurement, UML, Function Point, software design documentation.*

1. Introduction

Each projects, including IT project should begins with good planning and estimation on cost, schedule/time, human resource and activities. The activity of planning and estimating can be carried out by first measuring the size of the objects to be created or developed, thus measurement process become determinant in project valuation. Measurement can also describe attributes of the model or product being created [1].

One of the methods for measuring IT project, in this case, software development project is Function Point (FP) method which gives functional size of a software being developed. The function point measurement shows how far or how big functions given by the software to users. From the measures obtained using FP, project stake holders may have estimate on project cost, project duration and number of resources needed to complete the project. The FP method was introduced by Allan Albrecht and now it is kept updated by International Function Point User Group (IFPUG) [2].

IFPUG in its guidebook mentions that components to measure in FP analysis may not be easily obtained as users must understand and have deep analysis on the object in focus. This complexity introduces subjectivity of measuring process, leading to possibility of having different result for the same object being observed, as the measurement depends on the skill of the evaluators [2][3]. Function Point analysis includes two components: transactional function and data function.

2. Related Works

Discussions on the implementation of FP analysis have been poured into some papers, such as [4] and [5]. An example of measuring software volume based on FP is discussed in [4], while [5] proposed FP analysis using case-based reasoning (CBR) approach. Different approaches in mapping components of FP analysis for Object-oriented Software projects have also been discussed in [6], [7] and [8].

FP has been proven suitable to be implemented in the design stage as the beginning of software projects. In this case, UML design documents are taken as the basis for the calculation as tailored in [9], [10], [11], [12] and [13]. FP analysis also shows that it is easier to be adopted in the implementation of Functional User Requirements using UML compared to Cosmic-Full FP [14].

Measuring FP based on UML design, beside the abovementioned reason, has also more advantages such as: UML can accommodate various model of software development especially those of Object-Oriented Software and some UML diagrams provide interfaces easy to understand. Thus, measuring FP based on UML design may give the stakeholders better understanding on the software project. Use case diagram, in this approach, represents the transactional functions component ([7], [11],[14]), while class diagram represents the data function component which refers to logical files used in realizing the use case ([13], [11], [9], [10]). Rules and tools for performing FP analysis

on UML design obtained from Rational Rose were developed by [13]. Iorio in [11] adopted and improved the ideas and rules proposed by [7] and [13] in determining BFC (Base Functional Components) candidates of software documentation in the form of UML, while del Bianco in [10] proposed the FP calculation on UML design to reduce the difference in measurement between the parties conducting the software measurement and the software developers.
Considering the needs of end users and software developers in terms of FP measure, this research aims at designing, implementing and providing a system for making them easier in determining FP measure based on the existing UML design documents.

3. Research Methodology

3.1 Calculation and Automation Basis

Conforming to the aims of the research, the system is developed to measure the FP of a software project, based on UML design document, and to ease users by automating several manual calculation during FP calculation. The calculation or measurement refers to the IFPUG CPM version 4.3.1 in which functionality measure (stated in FP unit) is a sum of data function component plus transaction function component. The automation process of calculation is described in Table 1. Some terms appearing in Table 1 are explained as follows:

- uFP means unadjusted Function Point
- DET means Data Element Type, non-repetitive or non-recursive unique field read from file
- DET of DF refers to non-repetitive unique attribute coming from a data function, either in Internal Logical File or in External Interface File
- DET of TF refers to non-repetitive unique attribute needed to fulfill a transaction function, either in Internal Logical File (ILF) or in External Interface File (EIF)
- FTR means File Type Reference, a data function referred to or read by a transaction function
- RET means Records Element Type

From the Table 1, it is clear that the UML design diagrams taken as input for automating the FP measure in the system being developed can be use case diagram, class diagram or diagram describing relationships between both (use case and class diagrams).

3.2 Calculating Resources with Function Point

Function Point can be used for the determination of the resources on a software project. The resources are among others, concerning the time and effort required for the completion of a software project. The followings are formulas of resources calculation taken from [15] using only the size of the functionality (FP) for software development projects, regardless of platform and programming language used in the software development.

Project Work Effort (PWE)

PWE is the amount of work that shows the amount of time required for software development projects. The formula for determining PWE is:

$$PWE = C.\ (size^{E1}) \quad (1)$$

In which
PWE = Project Effort normalized to the developer team (Hours).
Size = software size (FP).
C = constant (23.25).
E1 = constant (0.814).
The median value MRE for the formula is 0.55

Project Duration

Project duration showed the amount of time required for the completion of software projects. The Project Duration can be calculated as

$$Duration = C\ .\ (size^{E1}) \quad (2)$$

In which
Duration = active time of software project (month)
Size = software size (FP).
C = constant (0.543).
E1 = constant (0.408).
The median value MRE for the formula is 0.41.

Speed of Delivery

Speed of delivery indicates the speed of project implementation by the entire team of software developers. The formula used has similar form

$$Speed = C\ .\ (size^{E1}) \quad (3)$$

In which:
Speed = processing speed for entire project team (FP/month)
Size = software size (FP).
C = constant (1.842).
E1 = constant (0.592).
The median value MRE for the formula is 0.40.

4. System Design and Implementation

4.1 System Architecture

The system for FP analysis is depicted as abstraction layer conforming to [16], as shown in Figure 1.
Presentation Layer is comprised of a set of user interfaces enabling user to access the system, while Application Logic layer contains a set of control classes responsible for

controlling actions to be carried out based on users' request. Domain Layer consists of entity classes connecting the system with its database and Database Layer is part where data are stored and manipulated through operations or methods existing in the entity classes.

4.2 Implementation

FPA system, based on the above architecture, is developed by implementing some modules. Among the main modules' features are:

a. Creating new project. A project is an entire process of calculating or measuring FP: input, calculating FP, generating report.
b. Calculating FP measure. Main input for the module is XMI document describing any UML design document. The input is then automatically parsed resulting FP components data. After the components are verified by users, the FP measure will be calculated. The FP measure is the aggregate between the measure of transaction function component and data function component obtained form the calculation based on [3].
c. FP Report Generation. The result of calculation will be presented as reports containing: project name, UML document used as input, XMI file name, time of execution, uFP measure and estimate of resources needed for the software project.

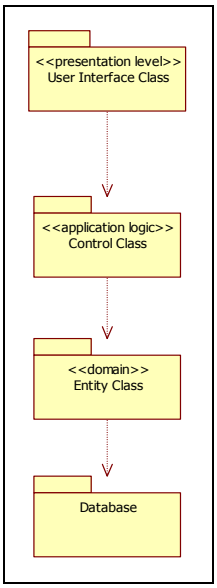

Fig. 1. Architecture of FPA System

Table 1 : Automation Process In The System For Calculating The FP Measure

FP calculation process		Calculation Basis		
		Use Case Diagram	Use case and Class diagram	Association between Use Case and Class Diagram
user identification		from actor or package	from actor or package	from actor or package
application boundary identification		from actor or package	from actor or package	from actor or package
transaction function (TF) identification		from use case	from use case	from use case
TF type determination		by users	by users	from the way how to refer logical file
TF Complexity	DET	TF is given complexity value "Average"	TF is given complexity value "Average"	from class attribute plus additional DET from user
	FTR			from association between use case and class diagram
DF identification		unknown	from class diagram, selected by user	from class diagram
DF type determination			by user	from the way DF is referred by TF
DF Complexity	DET		from class attribute	from class attribute
	RET		from multiplicity and dependency among classes	from multiplicity and dependency among classes
uFP calculation		uFP = FP_{TF}	uFP = FP_{TF} + FP_{DF}	uFP = FP_{TF} + FP_{DF}

5. Discussion

This section is divided into two main parts FP calculations: using case studies and the validation. In the first part, the Function Point calculation is performed on an appropriate software documentation (case study) using FP analysis system. The validation part is done by comparing the results of the FP analysis system with the IFPUG measure to check the validity of the calculation.

5.1 Resource Calculation Automation with FPA system

Case studies analyzed and calculated using FP analysis system is a simple system of Video Rental. FP analysis is performed on the use case and class diagram of the system Video Rental. Figure 2 is an example of video rental documentation in the form of a diagram illustrating the relationship between use cases and classes that are used for the realization of use cases.

FP calculation of diagram on Figure 2 is equal to 90 FP: 82 FP for actor Pak Joko and 8 FP for actor member. From Figure 3 and Figure 4 can be seen that the complexity of the transaction functions for both actors (Pak Joko and Member) vary, either high or low due to the adjustment based on the number of FTR and DET in the transaction function. Meanwhile, the complexity of the data function in both actors has complexity low according to the respective DET and RET. The measure 90 FP for rental projects is then used to estimate the resources using eq. (1), eq. (2) and eq. (3). The result of resource calculation can be seen in Figure 5.

Figure 5 describes calculation result of two different approaches (using platform PC and 4GL or without considering specific platform) for each measures (PWE, project duration and Speed of Delivery). The interpretation of the report is that using PC and 4GL the project is estimated to run faster.

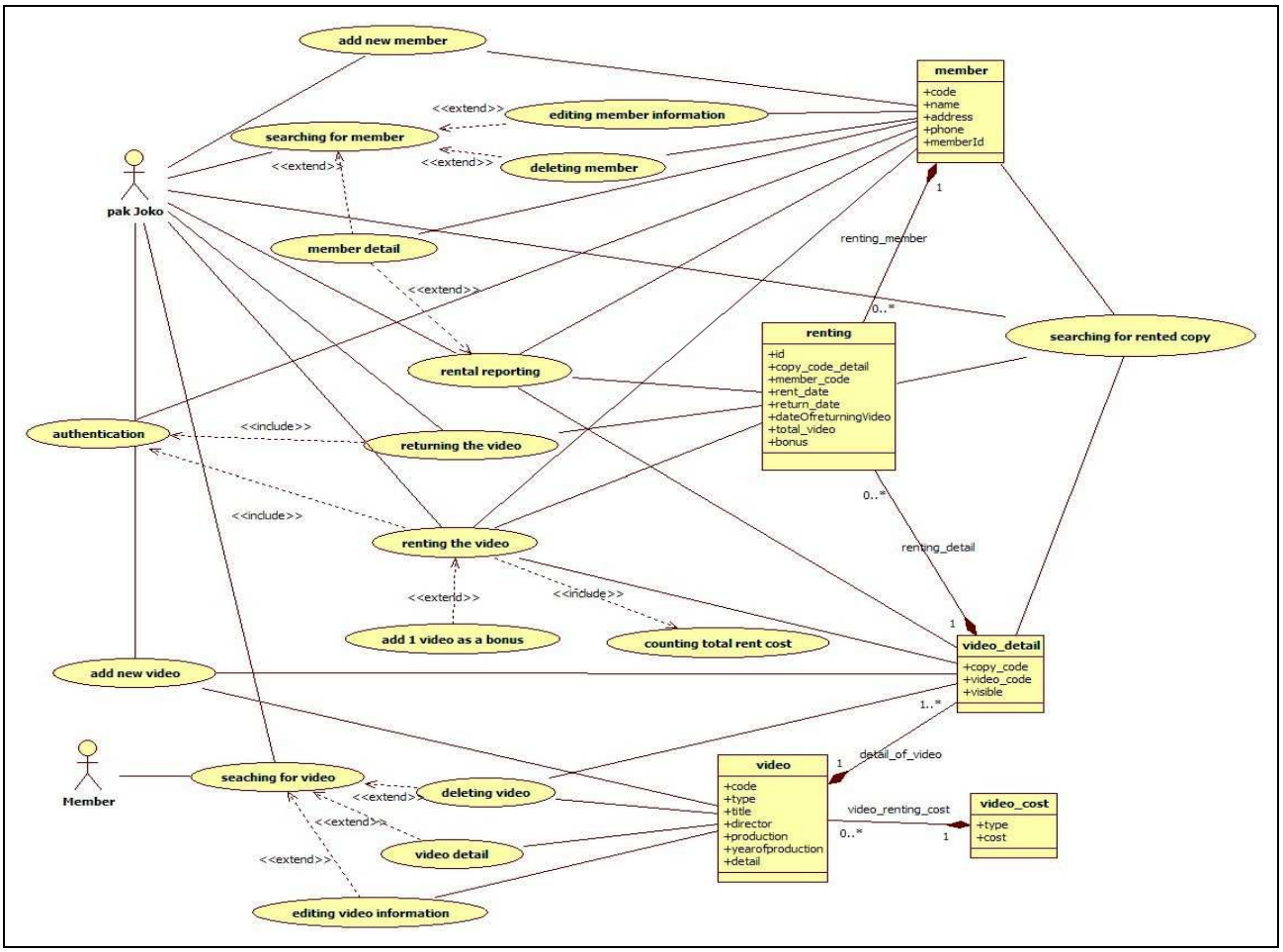

Fig. 2. Association between use case and class diagram of video rental system

Table 2: Comparison Between System Output With IFPUG Result

No	Case Study ID (from IFPUG)	Components							
		Type of function		Number of Ref. File (FTR/RET)		Number of DET		FP	
		System	IFPUG	System	IFPUG	System	IFPUG	System	IFPUG
1	SK 1-9	ILF	ILF	2	2	9	8	7	7
2	SK 1-15	ILF	ILF	1	1	4	4	7	7
3	SK 1-21	ILF	ILF	1	1	7	6 and 3	7	7
4	SK 1-29	ILF	ILF	2	1	7	6 and 3	7	7
5	SK 1-39	EIF	EIF	1	1	6	6	5	5
6	SK 1-43	EIF	EIF	1	1	2	2	5	5
7	SK 1-62	EIF	EIF	1	1	6	2	5	5
8	SK 2-73	EI	EI	1	1	5	6	3	3
9	SK 2-77	EI	EI	3	3	12	11	6	6
10	SK 2-85	EI	EI	2	1	15	11	4	3
11	SK 2-91	EI	EI	3	3	21	13	6	6
12	SK 2-105	EO	EO	4	4	15	5	7	5
13	SK 2-108	EO	EO	3	3	13	8	5	5
14	SK 2-131	EQ	EQ	1	1	11	6	3	3
15	SK 2-145	EQ	EQ	3	3	15	10	4	4
16	SK 2-159	EQ	EQ	1	1	4	4	3	3

FUNCTION POINT DETAIL

Application Boundary : pak Joko

Transactional Function (61FP)

No	Function	Type	DET	FTR	Complexity	uFP Size
1	Add new member	EI	5	1	Low	3
2	Renting the video	EI	23	3	High	6
3	Searching for video	EQ	10	1	Low	3
4	Add new Video	EI	10	1	Low	3
5	Searching for member	EQ	5	1	Low	3
6	Searching for rented copy	EQ	23	3	High	6
7	Rental Reporting	EO	23	3	High	7
8	Returning the video	EI	23	3	High	6
9	Add 1 video as a bonus	EI	18	2	High	6
10	Deleting video	EI	10	1	Low	3
11	Editing video information	EI	10	1	Low	3
12	Deleting member	EI	5	1	Low	3
13	Member Detail	EQ	5	1	Low	3
14	Editing member information	EI	5	1	Low	3
15	Video detail	EQ	10	1	Low	3
Total						61

Data Function (21FP)

No	Function	Type	DET	RET	Complexity	uFP Size
1	Member	ILF	5	1	Low	7
2	Video - Video_detail	ILF	10	2	Low	7
3	renting	ILF	8	1	Low	7
Total						21

Fig. 3. The results of the FP calculations based on the relationship between use case and class diagrams for the Actor Pak Joko

FUNCTION POINT DETAIL

Application Boundary : Member

Transactional Function (3FP)

No	Function	Type	DET	FTR	Complexity	uFP Size
1	Searching for video	EQ	7	1	Low	3
Total						3

Data Function (5FP)

No	Function	Type	DET	RET	Complexity	uFP Size
1	Video - Video_detail	EIF	7	1	Low	5
Total						5

Fig. 4. The results of the FP calculations based on the relationship between use case and class diagrams for actor Member

PROJECT ESTIMATION

Using ISBSG Regression Equation

Estimated from software size only in function points (IFPUG)

Project Size : 90 function points

Project Work Effort (New)	906	hours
Project Work Effort (New, PC, 4GL)	589	hours
Project Duration (New)	3.4	months
Project Duration (New, PC, 4GL)	2.6	months
Speed of Delivery (New)	26.4	FP/month
Speed of Delivery (New, PC, 4GL)	35.2	FP/month

Fig. 5. The results of calculations estimating the project resources with 90FP

PROJECT ESTIMATION
Using ISBSG Regression Equation
Estimated from software size only in function points (IFPUG)
Project Size : 65 function points

Project Work Effort (New)	695	hours
Project Work Effort (New, PC, 4GL)	448	hours
Project Duration (New)	3.0	months
Project Duration (New, PC, 4GL)	2.2	months
Speed of Delivery (New)	21.8	FP/month
Speed of Delivery (New, PC, 4GL)	29.5	FP/month

Fig. 6. The results of calculations estimating the project resources with 65FP

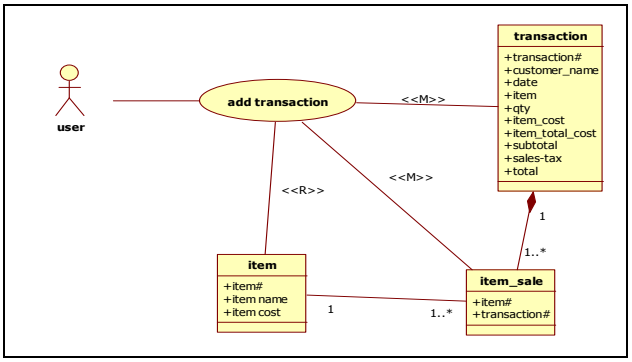

Fig. 7. Use Case and Class Diagram for the SK 2-85 assumed by the authors

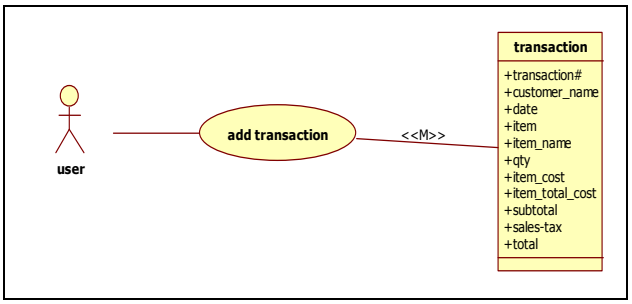

Fig. 8. Use Case and Class Diagram for the SK 2-85 assumed by IFPUG

The Figure 3, Figure 4 and Figure 5 are screenshots for a project whose input to FPA system is relationships between Use Case diagram and Class Diagram implementing the use cases. Evaluation and testing shows that measurement based on only Use Case Diagram, only Class diagram and relationships between both can lead to different results or FP values.
Using same scenario, calculation based on just use case diagrams or class diagram gives less value of measure. FP calculations for actor Member give measure 4FP, and for actor Pak Joko 61FP, giving total 65FP. Figure 6 shows the calculation report for 65FP.

5.2 Comparing FPA System Output against IFPUG Standard

It has been shown in the previous section that calculating FP became more handy with the aid the FPA system. The next step is to benchmark the accuracy of the FP result obtained from the system with the value according to IFPUG. The comparison is carried out by applying the FPA system to calculate the FPs of several scenarios included in IFPUG 4.3.1 CPM [3]. Table 2 is the summary of the comparison between system outputs and IFPUG results.
Table 2 shows that the result or output given by FPA system is sufficiently accurate; as among sixteen scenarios or UML diagrams provided in the IFPUG document, there are fourteen scenarios match with output of the system, while two of the scenarios (SK 2-85 and SK 2-105) have only slightly different FP values. This different calculation results come from the difference in perceptions regarding the delineation of data functions between analysts.

For example in the scenario SK 2-85, user requirement said that when an item is inserted then the price of goods (item cost) of such articles will appear automatically. This means that when the user starts to maintain a data item in a transaction file then the application will refer to the file item to pick up its price. Based on the requirement, we can draw a UML diagram to be transformed into XMI, and then fed into FPA system. In this case, the author assumed that at least two files or classes needed for the realization of the use case `add transaction`: the class `transaction` and class `items`. In addition, the relationship between class `transaction` and class `item` is many to many relationships and generate a class namely `item sale`. The UML assumed by author is shown in Figure 7 (resulting FP value of 4), while the UML assumed by IFPUG is shown in Figure 8 (resulting FP of 3).

6. Conclusion

After doing all the research stages, including developing and testing the FPA system, the following conclusions can be drawn:

1. In calculating FP using the system just developed, the more complete UML taken as input the more accurate the FP calculation results obtained.
2. The validation of the FP calculations results indicate that the FP calculations can be done automatically and quickly and with accurate results on the documentation of UML diagrams with certain conditions, such as:

a) Use cases drawn should qualify as an elementary process. Use case which is elementary process should be linked to a particular actor, or extend the use case that is connected to the actor.
b) The class diagram must be completely drawn including the attributes, types of dependencies and the multiplicity relationship between classes.
c) Use case and class relationship diagram must be equipped with a stereotype describing the type or way of reference (read only, read and calculating, or maintain).

3. Differences results of the FP calculation between software design and software product may occur due to differences in the number of DET. In software products, DET involved can be verified more precisely as it can be known from certain input data or output data involved in the fulfillment of certain functionality.

Acknowledgments

The authors would like to thank Indonesian National Education Ministry that provides the funding for this research and Department of Computer Science and Electronics of Gadjah Mada University which provides guideline and technical assistance for the research.

References

[1] M. Bundschuh and C. Dekkers, The IT Measurement Compendium : Estimating and Benchmarking Success with Functional Size Measurement. Springer-Verlag Berlin Heidelberg, 2008, e-ISBN 978-3-540-68188-5.
[2] D. Longstreet, "Function point analysis training course," online resource, last access: August 2011.
[3] I. F. P. U. Group, Function Point Counting Practices Manual release 4.3.1, 2010.
[4] Kusrini and M. Iskandar, "Pengukuran volume software berdasarkan kompleksitasnya dengan metode function point", Jurnal DASI, 2006.
[5] K. Mahar and A. El-Deraa, "Software project estimation model using function point analysis with cbr support", in Proceedings of IADIS Conference on Applied Computing, 2006.
[6] A. Chamundeswari and C. Babu, "An extended function point approach for size estimation of object-oriented software", in Proceedings of 3rd India Software Engineering Conference (ISECESE), India, Pebruary 2010.
[7] T. Fetcke, A. Abran, and T.-H. Nguyen, "Mapping the oo-jacobson approach into function point analysis", in Proceedings of the Tools-23: Technology of Object-Oriented Languages and Systems, ser. TOOLS '97. Washington, DC, USA: IEEE Computer Society, 1997, pp. 192–202. [Online].
[8] G. Giachetti, B. Marin, N. Condori-Fernandez, and J. C. Molina, "Updating oo-method function points", in Proceedings of the 6th International Conference on Quality of Information and Communications Technology. Washington, DC, USA: IEEE Computer Society, 2007, pp. 55–64.
[9] G. Cantone, D. Pace, and G. Calavaro, "Applying function point to unified modeling language: conversion model and pilot study", in Proceeding. 10th International Symposium on Software Metrics. Los Alamitos, CA, USA: IEEE Computer Society, 2004, pp. 280 – 291.
[10] V. del Bianco, C. Gentile, and L. Lavazza, "An evaluation of function point counting based on measurement-oriented models", in Proceedings of the International Conference on Evaluation and Assessment in Software Engineering 2008.
[11] T. Iorio, "Ifpug function point analysis in a uml framework", in Proceedings of SMEF, 2004.
[12] NESMA, "FPA applied to uml/use cases versi 1.0", 2008.
[13] T. Uemura, S. Kusumoto, and K. Inoue, "Function point measurement tool for uml design specification", in Proceedings of the 6th International Symposium on Software Metrics. Washington, DC, USA: IEEE Computer Society, 1999, pp. 62.
[14] K. G. van den Berg, T. Dekkers, and R. Oudshoorn, "Functional size measurement applied to uml-based user requirements," in Proceedings of the 2nd Software Measurement European Forum (SMEF2005), Rome, Italy, ser. Software Measurement European Forum (SMEF2005), March 2005, pp. 69–80.
[15] P. Hill and I. S. B. S. Group, Practical Software Project Estimation: A Toolkit for Estimating Software Development Effort & Duration. The McGraw-HilCompanies, Inc., 2010.
[16] S. Bennett, S. McRobb, and R. Farmer, Object-oriented systems analysis and design using UML. McGraw-Hill Companies, Inc., 2006.

Anie Rose Irawati is an active lecturer in the Department of Computer Science, University of Lampung, Indonesia since 2006. She receives her Master degree in Computer Science from Universitas Gadjah Mada, Indonesia in 2011.

Khabib Mustofa is an Assistant Professor at the Department of Computer Science and Electronics, Gadjah Mada University, Indonesia. He obtained his doctorate degree in Computer Science from The Institute for Software Engineering and Interactive System, Vienna University of Technology, Austria.