Estimated Measurement Quality Software On Structural Model Academic System With Function Point Analysis

Hetty Rohayani
Doctor of Computer Science Program
Bina Nusantara University
Jl. Kebon Jeruk No27, Jakarta.
Indonesia
hetty_mna@yahoo.com

Ford Lumban Gaol
Doctor of Computer Science
Bina Nusantara University
Jl. Kebon Jeruk No27, Jakarta.
Indonesia
fgaol@binus.edu

Benfano Soewito
Binus Graduate Program
Bina Nusantara University
Jl. Kebon Jeruk No27, Jakarta.
Indonesia
bsoewito@binus.edu

Harco Leslie Hendric Spits Warnars
Doctor of Computer Science
Bina Nusantara University
Jl. Kebon Jeruk No27, Jakarta.
Indonesia
shendric@binus.edu

***Abstract*— In the software development indispensable is the suitability and accuracy in determining the size or value of the software to fit the operation to be performed. A wide variety of calculation methods have been widely used to estimate the size of the software, one of which is by using Function Point Analysis (FPA). Volume calculation software based on a scale of complexity. Since the point of measurement is highly subjective, in order to maintain consistency and validity of the results, the method should be run by an experienced professional. This method is then applied by the authors to measure the complexity of academic information system STIKOM Dinamika Bangsa Jambi using structured modeling approach. Measurements were performed in this study consisted of depictions information system is built into the structure. Which is then analyzed by counting models Crude Function Points (CRP), the relative complexity of Adjustment Factor (RCAF), and then calculate the point function. From the results of calculations using the FPA to software quality measurement academic system STIKOM Dinamika Bangsa Jambi obtained value FP 166.32 is good. Function point value produced will be used by developers in determining the price and the cost of software systems to be built or developed.**

Keywords— Software Quality Measurement; Structural Model; Function Point Analysis; System

I. Introduction

The success of a software project is determined by many factors that are interconnected in a project. A project will be declared a success if all the existing requirements can be met, including the cost is not excessive (overflow), the schedule not exceeding the time limits specified. This can be done either by estimating the size of the volume of software systems that will provide accuracy the complexity and value of the product price on the software project being built. So that developers can plan resources, project costs, and the duration required precisely to build up a good software [1],[2].

Software development can be defined as the process of creating a new software to replace the old software as a whole or improve existing software. To be faster and more accurate in describing and developing software solutions, as well as the results easily developed and maintained, then the software development requires a specific methodology. Software measurement methodology is a process of organizing a collection of methods and notation conventions that have been defined to measuring the software [3]. The use of a methodology in accordance with the problems to be solved and meet the needs of users will produce a product engineering quality and well-maintained, and can avoid the problems that often occur as scheduling and cost estimation, software that does not comply with the wishes of users and so on.

For the developer, measuring the volume of useful software for resource planning, cost and duration necessary to build a software [3]. In addition, developers can also evaluate the quality of a product by comparing the volume of the system with the number of errors (error-count) done in software. Meanwhile, from a business perspective, the volume of software can be the basis for determining the value of the price of the software in question.

In the past many measures the volume of a software using a simple method that LOC (Lines Of Code) or KLOC (1000 Lines Of Code) [4],[5], which is a software engineering major measurement by counting lines of existing code. This technique has the properties he lacked, namely:

1. Relative to the language / programming tools and programmers coding style. LOC is highly dependent on the characteristics of programming tools used and programmers coding style. For example in the BASIC language code as follow:

```
a=a+1
```

only requires one line of code. Meanwhile, to get the same results in the PASCAL language the code is converted as follows:

```
program
x; var
        a: integer;
begin
        a:= a+1;
end.
```

which requires six lines of code. Also note the example of the difference of two script coding style following courses that lead to differences in LOC.

Code 1 (2 line)	Code 2 (4 line):

a:=a+1; b:=5;
if a=2 then a:=1;
	a:=a+1;
	b:=5;
	if a = 2 then
	a:= 1;

6. LOC can not be determined prior to completing the project development stages of implementation (coding). Therefore, the LOC can not be used to plan development process and also can be used to estimate the price of the product. The completion of the implementation phase is a phase that is too late to prepare a resource estimate.

The case of shortage then the desire to obtain a software volume measurement techniques that are not just based on the number of lines of programming code, but more towards something that can be measured early in the software development life cycle so that later came out the idea of method Function Point. In this study, the authors will discuss the use of the methods Function Analysis Point (FPA) to measure the estimated volume of the software system, the system of academic education Stikom Dinamika Bangsa Jambi will be compared to the estimated model using Entity Relation Diagram (ERD) based object model with the data flow Diagrams (DFD) based structured model.

2. LITERATURE REVIEW

a. Function Point Analysis (FPA)

FPA method is a part of The FSM (Functional Size Measurement) method was first introduced by Albrecht in 1979 as a method for measuring the amount of complexity and functionality in a software project [7],[8],[9],[10]. In the FPA procedure there are a variety of transactions, comprising the incoming and outgoing data to be processed on the system. Each transaction in the FPA will be mapped to the following models.

In the picture above can be seen, The Function Point Model consists of [6] :

1. External Input (EI)
   Functions that move data into the application without presenting data manipulation.
2. External Output (EO)
   Functions that move data to user and presents some data manipulation.
3. External Inquiries (EQ)
   Functions that move data to user without presenting data manipulation.
4. Internal Logical Files (ILF)
   The logic in the form of fixed data managed by the application through the use of External Input (EI)
5. External Interface Files (EIF)
   The logic in the form of fixed data used by the application but did not run in it.

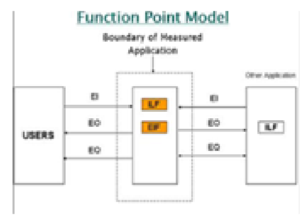

Fig 1: Function Point Method [11]

In the computation phase, each transaction is sorted by the amount of data that use. Logical transaction or file sorted based on the entities (called RET or Referenced Entity Types) and attributes (called DET or Data Entity Types). Functional transaction sorted based on the attribute numbers (DET), which moved out of the line and the numbering of logical transaction references. Then the whole categorized into 'low', 'average', or 'high' which each category given value as the value of Function Point (FP)[12],[13].

The stages are there in determining the function point is [14],[15],[16].

Step 1: Calculating The CFP (Crude Function Points)

The number of functional components of the system were first identified and followed to evaluated the complexity of quantization weight of each component. Weighting was the summed and become the number of CFP. CFP calculation involves five types of software system components following :

1. The number of input applications
2. The number of output applications
3. The number of online query applications.
4. This application related to query against the data stored
5. The number of logical files or tables which involved
6. The number of external interfaces

An output or input interface that can be connected to the computer through data communications, CDs, floppy disks, etc. Then given a weighting factor to each of the above components based on its complexity. The table below is an example of the weighting blank:

Table 1. Blank of CFP Calculation [9]

Software	Complexity Level									Total
System	Simple			Average			Complex			CFP
Component	C	W	P	C	W	P	C	W	P	
Inputs	-	3	-	-	4	-	-	6	-	-
Outputs	-	4	-	-	5	-	-	7	-	-
Online Queries	-	3	-	-	4	-	-	6	-	-
Logical Files	-	7	-	-	10	-	-	15	-	-
External Interfaces	-	5	-	-	7	-	-	10	-	-
Total CFP										-

Step 2: Calculating the complexity of transcription factors of RCAF (Relative Complexity Adjustment Factor) for the project. RCAF is to calculate the complexity assessment of software system from several characteristics of subject. Rating scale from 0 to 5 is given to each subject that most affect the development effort required. Example of RCAF assessment form can be seen as follows :

Table 2. RCAF Assessment Form [9]

No	Subject	Value				
1	The level of recovery reliabilty complexcity	1	2	3	4	5
2	The level of data communication complexcity	1	2	3	4	5
3	The level of dIstributed processing complexity	1	2	3	4	5
4	Level of the need for performance complexity	1	2	3	4	5
5	The level of operating environment demand	1	2	3	4	5
6	The level of developer knowledge needs	1	2	3	4	5
7	The level of updating the master file complexity	1	2	3	4	5
8	The level of rotallection complexity	1	2	3	4	5
9	The level of input, output, online query and file application complexity	1	2	3	4	5
10	The level of data processing complexity	1	2	3	4	5
11	The improbability level of reuse code	1	2	3	4	5
12	The level of custumer organization variation	1	2	3	4	5
13	The expert of possible changes	1	2	3	4	5
14	Level of the ease of use demand	1	2	3	4	5
	Total RCAF					

Step 3: Calculating Function Points by the formula [9].

$$FP = CFP \times (0.65 + 0.01 \times RCAF) \quad (1)$$

Values for the function point software system is then calculated based on the results of phase 1 and 2 are put into the formula.

b. Structural Model

In DFD, there are some important components which is component as terminators in a box of a source or destination, the components in the form of a circle of as a description of the activities that will be/are being implemented consisting of input and output, data store component in the form of two horizontal lines are lined up as an image of computerized storage, and data flow components depicted in the form of an arrow as data transfer from one section to another system [17].

In ERD, the modeling made up of several components, namely the entities, attributes, relationships, cardinality ratio and constraint participant. An entity is an object that will store, produce information. Attributes are the characteristics of the entity or relationship that provides a detailed explanation of the entity or relationship. Relationship is the relationship between one or more entities. The cardinality ratio is a limit to the number of connection one entity to another entity. While the constraint participant clarify whether the existence of an entity depends on its relationship with other entities [18].

3. RESULT AND DISCUSSION

In this research, measurement of software quality system of academic information STIKOM Dinamika Bangsa Jambi, where the system will provide services such as filling Study Plan Card or KRS online, structured approach in a paper done by using the model of data flow diagram (DFD) and Entity Relationship diagram (ERD). DFD is a diagram to describe the system as a network function processes are connected to each other by data flow. While ERD is a model of the logical structure of the database lecture schedule, and check the value of the program.

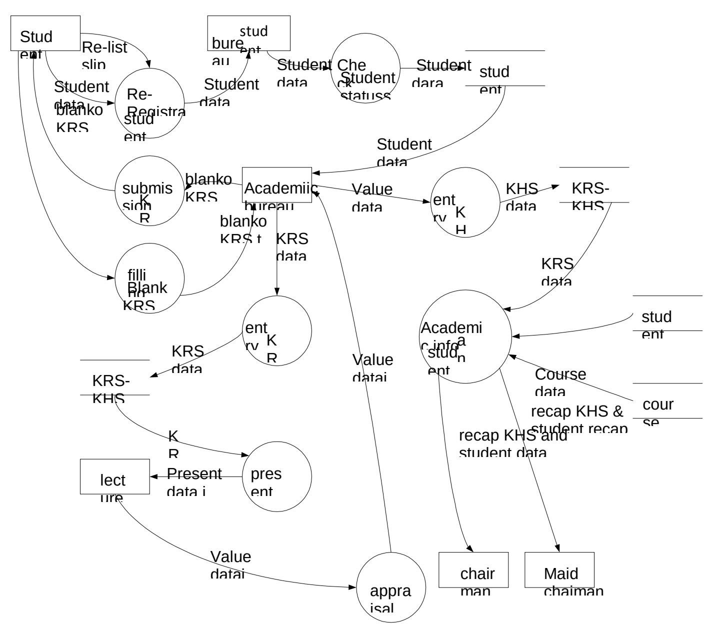

Fig 2. Data Flow Diagram of Academic System STIKOM

Step 1 : Calculate Function Point Crude

Based on the image below will be shown an example of calculating the value of function point software system for the proposed academic information STIKOM Dinamika Bnagsa Jambi. From the image data flow diagram in Figure 2, it can be determined the values needed to calculate the CFP, which is the amount of each component, as follows:

- The number of input aplication: 4
- The number of output aplication: 3
- The number of online query: 9
- The number of logic file: 6
- The number of external interface: 0

After that, re-evaluated to sort each component based group degree of complexity (simple, medium or complex) by a professional to get the value of the CFP. In this experiment, the results of these evaluations are presented in Table 3 , which after calculation resulted in a total value of CFP = 154.

b. Class Diagram

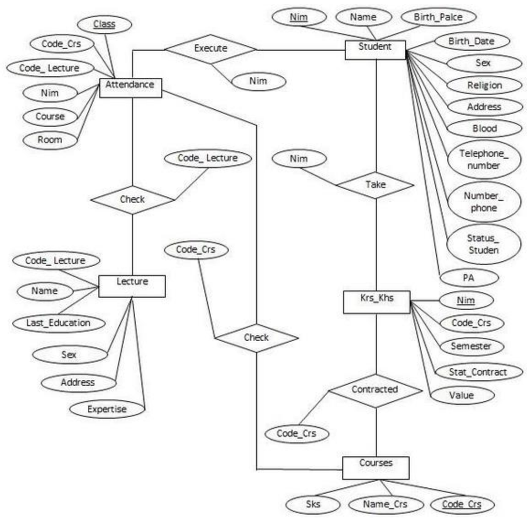

Fig 3. Entitas Relation Diagram of Academic System STIKOM

Table 3. CFP Calculation of Academic System STIKOM based on Structural Model

Software	Complexity Level									Total
System	Simple			Average			Complex			CFP
Component	C	W	P	C	W	P	C	W	P	
Inputs	-	3	-	1	4	4	3	6	18	22
Outputs	-	4	-	-	5	-	3	7	21	21
Online Queries	3	3	9	-	4	-	6	6	36	36
Logical Files	-	7	-	3	10	30	3	15	45	75
External Interfaces	-	5	-	-	7	-	-	10	-	-
Total CFP										154

Step 2: Calculate the relative complexity adjustment factor(RCAF).

The next step is to evaluate the complexity characteristics of STIKOM Dinamika Bangsa Jambi and calculation of the RCAF. RCAF calculation results in this study are described in Table 4, there is shown a summation to the 14th element / factor adjustment relative complexity of generating value 43.

Table 4. RCAF Calculation of Academic System STIKOM Dinamika Bangsa

No	Subject	Value
1	The level of recovery reliabilty complexcity	5
2	The level of data communication complexcity	4
3	The level of dIstributed processing complexity	3
4	Level of the need for performance complexity	4
5	The level of operating environment demand	3
6	The level of developer knowledge needs	3
7	The level of updating the master file complexity	2
8	The level of rotallection complexity	3
9	The level of input, output, online query and file application complexity	2
10	The level of data processing complexity	3
11	The improbability level of reuse code	2
12	The level of custumer organization variation	3
13	The expert of possible changes	3
14	Level of the ease of use demand	3
Total RCAF		43

Then the function point of academic system STIKOM Dinamika Bangsa Jambi based on structured modeling can be calculated :

$$FP = CFP \times (0.65 + 0.01 \times RCAF)$$

$$= 154 \times (0.65 + 0.01 \times 43)$$

$$= 166.32$$

CFP calculation results can be used as the basis for calculating large projects such as the time required for project development, project costs, the level of resources required and so on.

4. CONCLUSION

From the description and result of the experiments carried out a number of conclusions :

1. Method of function point cab be used as an alternative to calculate the volume of software based on its complexity.
2. The function point method has several advantages compared LOC namely:
   a. Does not depend on the programming language used by programmers.
   b. It does not depend on the style of programming a programmer.
   c. Can be determined early in the software development.
3. The use of function point method has the disadvantages of requiring the intervention of an experienced professional for calculation is done in a very subjective analysis.
4. Since the calculations are done a lot based on the description of the data processing, methods of function point should be supported by additional data to strengthen the approximate volume of a software system that will be generated.

REFERENCES

[1] J. G. Borade, “International Journal of Advanced Research in Software Project Effort and Cost Estimation Techniques,” vol. 3, no. 8, pp. 730–739,

2013.
[2] F. L. Gaol, B. Soewito, H. Leslie, and H. Spits, “Object Oriented Metrics to measure the quality of software study case Request online System,” 2016.
[3] M. Lee and T. Chang, “Software Measurement and Software Metrics in Software Quality,” vol. 7, no. 4, pp. 15–34, 2013.
[4] M. Sadiq, M. Asim, J. Ahmed, V. Kumar, and S. Khan, “Prediction of Software Project Effort Estimation : A Case Study,” vol. 1, no. 1, pp. 37–43, 2011.
[5] K. P. Srinivasan, “Unique Fundamentals of Software Measuremant And Software Metrics,” vol. 7, no. 4, pp. 29–43, 2015.
[6] D. Pratiwi and D. Pratiwi, “Implementation of Function Point Analysis in Measuring The Volume Estimation of Software System in Object Oriented ... Implementation of Function Point Analysis in Measuring the Volume Estimation of Software System in Object Oriented and Structural Model of Academic System,” no. August 2016, 2013.
[7] C. Gencel, M. East, A. Prof, and M. East, “Conceptual Differences Among Functional Size Measurement Methods.”
[8] N. Choursiya and R. Yadav, “An Enhanced Function Point Analysis (FPA) Method for Software Size Estimation,” vol. 6, no. 3, pp. 2797–2799, 2015.
[9] A. R. Irawati and K. Mustofa, “Measuring Software Functionality Using Function Point Method Based On Design Documentation,” vol. 9, no. 3, pp. 124–130, 2012.
[10] G. Xunmei, S. Guoxin, and Z. Hong, “The Comparison between FPA and COSMIC-FFP,” pp. 109–117.
[11] T. Fetcke, A. Abran, T. Nguyen, and M. Q. C. Hc, “Mapping the OO-Jacobson Approach into Function Point Analysis.”
[12] M. Heri and A. Živkovi, “A Method for Calculating Acknowledged Project Effort Using a Quality Index,” vol. 31, pp. 431–436, 2007.
[13] T. Arnuphaptrairong, “The Development and Achievements of Software Size Measurement,” vol. I, 2012.
[14] W. Wikusna, “InSearch, Universitas Informatika dan Bisnis Indonesia - 1 of 9.,” pp. 1–7.
[15] K. Kusrini, “Pengukuran Volume Software berdasarkan Kompleksitasnya dengan Metode Function Point,” no. September, 2015.
[16] R. K. Hapsari and M. J. Husen, “Estimasi Kualitas Perangkat Lunak Berdasarkan Pengukuran Kompleksitas Menggunakan Metrik Function Oriented,” pp. 425–434, 2015.
[17] A. Elbakush, “Functional Modeling with Data Flow Diagrams.”
[18] H. K. Al-masree, “Extracting Entity Relationship Diagram (ERD) From Relational Database Schema,” vol. 8, no. 3, pp. 15–26, 2015.