Case Study Evaluations for a Function Point Counting Improvement for Object-Oriented Projects

José Antonio Pow-Sang[1,2], Arturo Nakasone[1], Ricardo Imbert[2], and Ana María Moreno[2]
[1] Departamento de Ingeniería, Pontificia Universidad Católica del Perú, Peru
{japowsang, arturo.nakasone}@pucp.edu.pe
[2] Facultad de Informática, Universidad Politécnica de Madrid, Spain
{rimbert, ammoreno}@fi.upm.es

Abstract. Since the introduction of object-oriented (OO) development in industrial practice, many Function Point (FP) technique adaptations have been introduced to improve software size estimation in these kinds of projects. Current research work only deal with OO modifications to the previous version of the FP Counting Practices Manual (4.1). In this paper, we propose the use of the composition relationship analysis in classes to improve the rules included in FP Counting Practices Manual 4.2.1 for Internal Logic Files (ILF) and External Interface Files (EIF) identification. We also show the results obtained by applying our proposal in six case studies performed by practitioners and comparing against the results we obtained with undergraduate students. These results have proved to be at least equal in accuracy and consistency to the original FPA technique.

Keywords: Function Points, Object Oriented, UML, Conceptual Model.

1 Introduction

Function Point (FP) [11] [12] is a software measurement technique created by Allan Albrecht for IBM [3], and has gradually become a sounder alternative to other popular size metrics methods, such as source lines of code (SLOC), making it one of the most widely used techniques.

With the diffusion of the Unified Modeling Language [19], promoted by the Object Management Group, many object-oriented approaches to calculate function points have been proposed. Unfortunately, they do not consider some important specifications included in UML, such as the composition relationship between classes. For this reason, we propose in this paper an approach to calculate Logic Files (ILF and EIF) from an analysis class diagram that makes use of composition relationships. We have also tested our approach against the standard Function Points Counting Practices Manual, version 4.2.1 [12] proposed by the International Function Points User Group (IFPUG) obtaining interesting and promising results.

The rest of the paper is organized as follows: Section 2 describes the related work in the FP measurement technique area, Section 3 details our proposed rules to identify logical files, Section 4 presents the background scenario for the empirical study;

Section 5 shows the obtained results for each case study; Section 6 discusses those results. Finally, a summary and our plans for future research will conclude our paper.

2 Related Work

In order to cope with object-oriented software measurement, several methods to calculate FP are being promoted and used. These methods reformulate the IFPUG rules in terms of OO concepts to facilitate the function points counting process. The final result of the count using these kinds of techniques is similar to what is obtained by directly applying IFPUG Function Point Analysis (FPA). Fetcke [9] defined rules for mapping the OO-Jacobson method [15] to concepts from the IFPUG Counting Practices Manual 4.0 and the results obtained from three case studies have confirmed that these rules can be applied in a consistent way. Uemura et Al. [20] proposed FPA measurement rules for design specifications based on UML (Unified Modeling Language), developing a FP measurement tool. Cantone et Al. [6] and Caldiera et Al. [5] defined rules to map OO concepts to FPA and performed pilot studies to demonstrate the feasibility of their approaches. Finally, Abrahao et al. [1] [2] present a FP-based method called OOmFP and its evaluation through an empirical study.

All of the proposals described above define rules of mapping OO to concepts from older versions of IFPUG Counting Practices Manual (CPM). Although Abrahao considers composition relationship, some mapping rules to calculate Logic Files (LF) are not in accordance with the last IFPUG CPM (last version is 4.2.1).

Moreover, the majority of the proposals presented above calculate FP-Logical Files (LF) from a class diagram, but they only consider the aggregation relationship and not the composition relationship. UML [19] defines composite aggregation or composition as a stronger form of aggregation, which requires that a part instance be included in at most one composite at a time and that the composite object has sole responsibility for the disposition of its parts. In summary, composition presents a stronger dependence relationship than aggregation. Our approach includes both aggregation and composition relationship in order to identify files and its record element types (RET).

3 Rules to Identify Logical Files

The input for our proposed rules is the analysis class diagram included in the Jaaksi's method [14] or the domain model mentioned by Larman [17]. In this model transformation, we are considering the following relationships among classes: association, aggregation and composition. Generalization and association class are not included. Neither do we take into account the difference between ILF and EIF. Our rules only deal with the identification of LF and its number of RETs.

- **Rule 1**. Classes that are connected by a composition relation can be mapped together to one LF with 2 RETs. For example, in Fig. 1 you must consider one LF and 2 RETs.

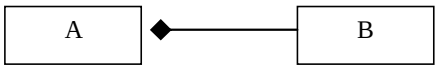

Fig. 1. Example of a composition relation

- **Rule 2.** If there are three classes A, B, and C and two of them (A and B) are connected by a composition relation as shown in Fig. 2, they must be mapped together to one LF for the composition relation with 2 RETs and another LF for class C.

Fig. 2. Example of a composition and association relations

- **Rule 3.** If there are two classes, A and B, which are connected through an association or aggregation relation, and neither of them is connected by a composition one to a third class, one must follow the indications shown in Table 1. The table is an adaptation for OO from IFPUG CPM 4.2.1 [12].

Table 1. Rules to identify LF from classes without composition relationships

Multiplicity A	Multiplicity B	When this condition exists	Then Count as LFs with RETs and DETs as follows:
0..*	0..*	A and B are independent	2 LFs
0..1	0..*	A and B are independent	2 LFs
1	1..*	If B is independent of A	2 LFs
		If B is dependent on A	1 LFs, 2 RETs
1	0..*	If B is independent of A	2 LFs
		If B is dependent on A	1 LFs, 2 RETs
0..1	1..*	If A is independent of B	2 LFs
		If A is dependent on B	1 LFs, 2 RETs
0..1	0..*	A and B are independent	2 LFs
1	1	A and B are dependent	1LF, 1RET
0..1	0..1	A and B are independent	2 LFs
1..*	1..*	If B is independent of A	2 LFs
		If B is dependent on A	1 LFs, 2 RETs
1..*	0..*	If B is independent of A	2 LFs
		If B is dependent on A	1 LFs, 2 RETs

4 Experimental Design

For our case study scenario, we have considered the experimental software engineering suggestions made by Juristo & Moreno [16]. Our experiment is similar to one presented by Abrahao et al. [1] [2], and its goal is to empirically corroborate which method provides the best functional size assessment to identify logical files.

Using the Goal/Question/Metric (GQM) template for goal-oriented software measurement [4] we defined this experiment using the following parameters:

- **Analyze**: LF measurement
- **For the purpose of**: Evaluating our approach against the one proposed by the IFPUG CPM 4.2.1
- **With respect to:** Their accuracy
- **From the point of view of:** The researcher
- **In the context of:** undergraduate and graduate students.

The formulated research question was: Does our approach produce accurate measurements of LF at least equal to those found in the IFPUG FPA?

4.1 Variables Selection

Our independent variable was the method used by subjects to size a case study, and our dependent variable was the accuracy: the agreement between the measurement results and the true value.

To obtain a "true value" for comparison, we took similar case studies included in the IFPUG CPM 4.2.1.

4.2 Students Who Participated in the Experiment

We selected a within-subject design experiment; in other words, the students had to use both our method and the IFPUG CPM 4.2.1 method to determine LF for each case study. The subjects were randomly assigned to either one of two groups using the counterbalancing procedure with equal number of participants in each group. The methods were applied in a reverse order.

- Group 1: Our approach first and then the IFPUG CPM 4.2.1 method.
- Group 2: The IFPUG CPM 4.2.1 first and then our approach.

The undergraduate students who participated in the experiment were fourth year students of the Informatics Program at Pontificia Universidad Católica del Perú (PUCP) that were enrolled in the Spring '06 Software Engineering course. Table II presents a summary of the undergraduate students' knowledge and experience at the beginning of the experiment. The majority of the courses in the Informatics program at PUCP focus on software projects as applications of theoretical concepts, but they do not demand the utilization of estimation and planning techniques during their development.

The practitioners were students of the Postgraduate Diploma in Software Engineering at PUCP in 2006. Students had at least two years of experience in software projects and, although they used OO development tools at work, most of them were not used to applying OO analysis and design techniques. For this reason, these students had to take an OO analysis and design course previous to the elaboration of the experiment.

4.3 Materials and Case Studies

The materials used in the experiment are:

- Description of Case studies.
- Form to fill in the number of LF and its RETs for each case study.
- A questionnaire to know student's opinion about which technique was easier to apply.
- A summary of Coad's patterns [8]. We explained these patterns to the undergraduate and graduate students in previous courses, so they can elaborate their diagrams correctly.

Table 2. Knowledge and experience that the undergraduate students possessed at the beginning of the experiment**.**

Characteristic	Knowledge and/or Experience
Programming Languages / Programming Environment	Java, C#, Pascal, C and Prolog.
Database Modeling Techniques	Entity Relationship Diagram, IDEF 1X
Analysis and Design Techniques	Structured and Object-oriented
Project Management	Experience in managing developing projects that included short software programming projects (Work Team of 3 or 4 students). No previous planning and estimation experience.

The descriptions of the six case studies with their analysis class diagrams are the following:

- *Case Study 1*: The objective was to develop a Sales System that automates the registration of customers and the data of their invoices. The information of the client is: client code, name, address, and phone number. The information of the invoice is: number of the invoice, date, and total amount. In this case, clients without invoices and invoices without clients can exist in the system.

Fig. 3. Class Diagram for Case Study 1.

- *Case Study 2*: The objective was to develop a Sales System that registers invoices only. The information of the invoices is: number of the invoice, date, tax amount, and total amount. Additionally, the detail of the invoice includes: line number, product description, product quantity, product unit price and subtotal. The system also allows the registry of the types of products that are sold and their information: code, description and price.

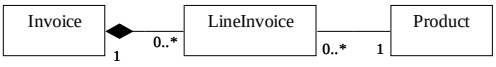

Fig. 4. Class Diagram for Case Study 2**.**

- *Case Study 3:* The objective was to develop a system that registers universities and their students. The information of the universities is: code, name, address, web page and telephone number. Students' information include: code, name, address, e-mail and telephone number. The system allows universities without students and students without universities. Students can belong to more than one university.

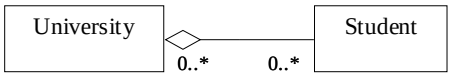

Fig. 5. Class Diagram for Case Study 3.

- *Case Study 4:* The objective was to develop a system that allows the registration of project plans and their activities. The system allows a project plan to exist without defined activities, but an activity must always be associated to a project plan exclusively.

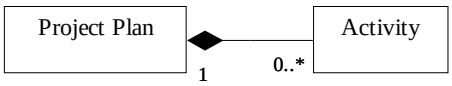

Fig. 6. Class Diagram for Case Study 4.

- *Case Study 5:* The objective was to develop a system that allows the registration and maintenance of CD information which is defined by its code, title and duration. Information on each track of the CD includes: number of track, song name, artist and duration.

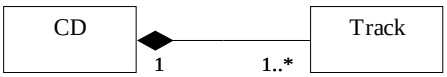

Fig. 7. Class Diagram for Case Study 5.

- *Case Study 6:* The objective was to develop a system that allows the registration of customer orders. Information for each order is: number, date and customer name. Additionally the detailed information for the order contains: line number, product code, product description and product quantity. The system also allows the registration of product information sold by the company, such as code and description.

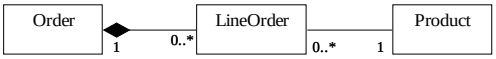

Fig. 8. Class Diagram for Case Study 6.

Further details of the case studies and used instruments can be found at:
http://macareo.pucp.edu.pe/japowsang/pf/composition.html

4.4 Tasks Performed in the Experiment

Table 3 shows the tasks carried out in the session by the students

Table 3. Tasks of the session carried out by the students

Number of Task	Group 1	Group 2
1	Reception of the case studies	
2	Reception of material with the explanation of our approach, Coad's patterns and form to fill number of LF and its RETs	Reception of material with the explanation of IFPUG CPM and form to fill number of LF and its RETs
3	Elaboration of analysis class diagrams and identification of LF	Elaboration of E-R diagrams and identification of LF
4	Delivery of completed forms back with the results	
5	Reception of material with the explanation of IFPUG CPM and form to fill number of LF and its RETs	Reception of material with the explanation of our approach, Coad's patterns and form to fill number of LF and its RETs
6	Elaboration of E-R diagrams and identification of LF	Elaboration of analysis class diagrams and identification of LF
7	Delivery of completed forms back with the results and reception of questionnaire	
8	Delivery of questionnaire	

The session lasted approximately one hour and the students performed forty and five minutes on average in all of the tasks. Although, our study did not include a timing analysis for each technique, we could observe that both groups used almost the same time to identify LF using both techniques. In addition, it is important to mention that we informed the students that the purpose of the questionnaire was to know their honest opinion about which technique was easier to apply.

5 Results

For each case study, we graded it with "1" (one) if the student correctly identified the number of LF and RET and "0" (zero) if he or she did it incorrectly.

Subsections 5.1, 5.2 and 5.3 present quantitative results and subsection 5.4 shows the results obtained from the questionnaire.

5.1 Undergraduate Students Results

Acknowledging the advantages of utilizing students in experiments [7], Table 4 presents the detailed results for each undergraduate student. For each case study, the first column shows the results obtained with the IFPUG CPM 4.2.1 technique and the second one with our proposal. Results presented in Table 4 were previously published in [18]

Table 4. Results obtained with undergraduate students

Student	Case Study											
	1		2		3		4		5		6	
1	0	1	1	0	1	1	0	0	0	0	1	0
2	1	1	1	1	1	1	1	1	1	1	1	1
3	1	1	1	1	1	1	1	1	1	1	1	1
4	1	1	1	1	1	1	1	1	1	1	1	1
5	1	1	0	1	1	1	1	1	1	1	0	1
6	1	1	0	1	1	1	1	1	1	1	0	1
7	1	1	0	1	0	1	1	1	0	1	0	1
8	1	1	0	1	1	1	1	1	1	1	0	1
9	1	1	1	1	1	1	1	1	1	1	1	1
10	1	1	1	1	1	1	1	1	0	1	1	1
11	1	1	1	1	1	1	1	1	1	1	1	1
12	1	1	1	1	1	1	1	1	1	1	1	1
13	1	1	1	1	1	1	1	1	1	1	1	1
14	1	1	1	1	1	1	1	1	1	1	1	1
15	1	1	1	1	1	1	1	1	1	1	1	1
16	1	1	1	1	1	1	1	1	1	1	1	1
17	1	1	1	1	1	1	1	1	0	1	1	1
18	1	1	1	1	1	1	0	1	0	0	1	1
19	1	1	1	1	1	1	1	1	1	1	1	1
20	1	1	1	1	1	1	1	1	1	1	1	1
21	1	1	1	1	1	1	1	1	1	1	1	1
22	1	1	0	1	1	1	0	1	0	1	0	1
23	1	1	1	1	1	1	1	1	1	1	1	1
24	1	1	1	1	1	1	1	1	1	1	1	1

A significance level of 0.05 was established to statistically test the obtained results. Since these results follow a non-normal distribution, the paired samples t-test could not be used and a non-parametric alternative was selected: the Wilkoxon signed rank test. The statistical hypotheses formulated to test both techniques are:

- H0: The distribution of the two samples is not significantly different.
- Ha: The distribution of the IFPUG CPM 4.2.1 sample is shifted to the left of the distribution of our proposed FPA sample.

Table 5. Wilkoxon signed rank test results for undergraduate students

Variable	Result
Observations	144
V	269.0
Expected value	1345.0
Variance (V)	50732.5
p-value (one-tailed)	<0.0001
Alpha	0.05

Since the computed p-value is lower than the significance level alpha=0.05 as shown in Table V, we can reject the null hypothesis H0 and accept the alternative hypothesis Ha. It means that we can empirically corroborate that our proposal produces more accurate assessments than the IFPUG CPM 4.2.1 approach.

5.2 Graduate Students Results

Eighteen students participated in the experiment. All of the graduate students, except one (he performed all case studies correctly using our approach, but only 2 case studies correctly using IFPUG), performed correctly the case studies using both techniques.

For these results, the same statistical test as the one utilized for undergraduate students results was used. The statistical hypotheses formulated to test both techniques are:

- H0: The distribution of the two samples is not significantly different.
- Ha: The distribution of the IFPUG CPM 4.2.1 sample is shifted to the left of the distribution of our proposed FPA sample.

Since the computed p-value is higher than the significance level alpha=0.05 as shown in Table VI, we cannot reject the null hypothesis H0 and reject the alternative hypothesis Ha. It means that we can not empirically corroborate that our proposal produces more accurate assessments than the IFPUG CPM 4.2.1. approach.

Table 6. Wilkoxon signed rank test results for graduate students

Variable	Result
Observations	144
V	0.0
Expected value	285.0
Variance (V)	0.0
p-value (one-tailed)	1.0
Alpha	0.05

Due the results obtained in Table 6, we changed the alternative hypothesis. The new Ha was *the distribution of the two samples is significantly different*. Using the information showed in Table VI, we still cannot reject H0, which empirically corroborates that our proposal and the IFPUG CPM 4.2.1 approach produces the same accurate assessment.

5.3 Comparison of Quantitative Results

From what it is observed in subsections 5.1 and 5.2, there is a difference in results from both kinds of students. Some of the reasons for those differences are:

Undergraduate students knew about structured techniques, but they were more familiar with OO techniques because they participated in OO software development projects before the experiment. They obtained better results with our proposal because they had more experience with OO approaches.

Many of the graduate students apply only structured techniques at work, although they use OO development tools. Therefore, their experience with OO techniques was not as sound as the experience of undergraduate students (before the experiment they were required to take an OO analysis and design course). Due to their experience in structured techniques, graduate students obtained better results using IFPUG CPM 4.2.1 than undergraduate students.

Based on the results obtained with undergraduate and graduate students, we can conclude that our proposal produces at least the same accurate assessment as that of the IFPUG CPM 4.2.1.

5.4 Questionnaire Results

As can be seen from Table 3, students filled a questionnaire about the techniques used in the experiment. Two multiple-choice questions (results are presented in figures 9 and 10) and a comment section were included.

Fig. 9 shows the results of the question: Rules regarding composition (technique with classes) facilitate LFs and RETs identification compare to technique without classes? They clearly show that all of the undergraduate students and most of the graduate students consider that our proposal facilitates LF identification. It can be observed that 16.7% of graduate students think both techniques are the same, due to their experience with structured techniques at work.

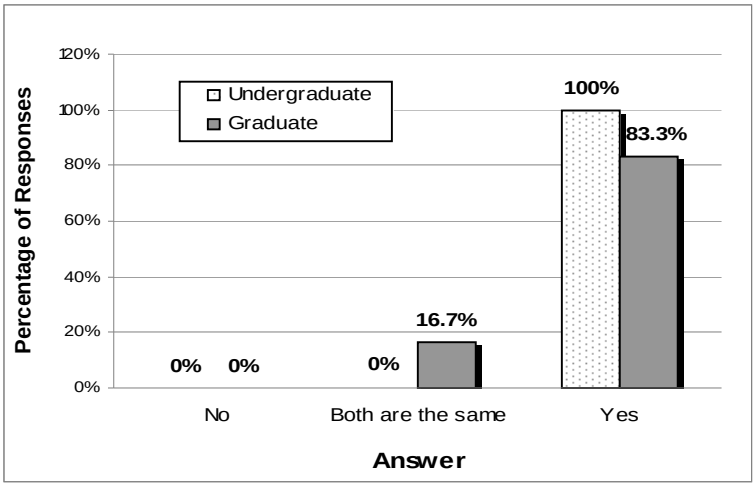

Fig. 9. Results of the question "Rules of our proposal facilitate LFs and RETs Identification?"

Fig. 10 presents the results of the question *The technique with classes is easier to apply than the technique without classes?* From these results, it is shown that most of the students (graduate and undergraduate) think that our proposal is easier to apply than the IFPUG CPM 4.2.1 approach.

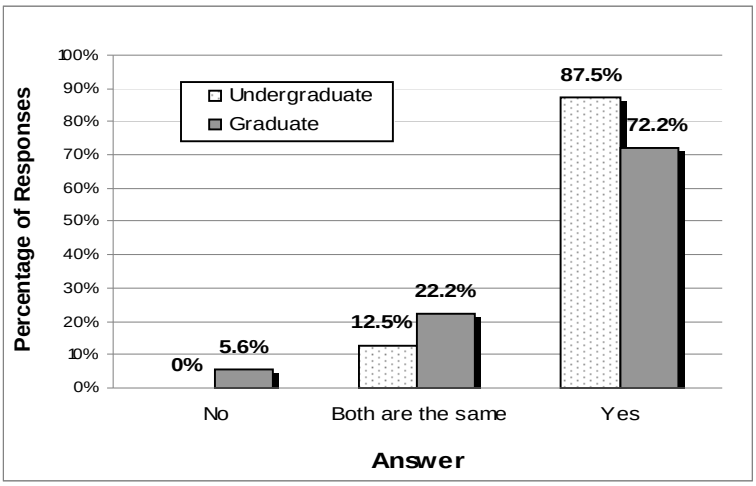

Fig. 10. Results of the question "The technique with classes is easier to apply than the technique without classes?"

As it can be observed from Figures 9 and 10, the questionnaire results do not disagree with the quantitative results presented in previous subsections. Few students

wrote comments, but all of them confirmed the results obtained in the multiple-choice questions.

6 Discussion

In this section, we discuss various threats to the validity of the empirical study and the way we tried to alleviate them.

6.1 Threats to Construct Validity

The dependent variable (accuracy) that we used is proposed in the ISO/IEC 14143-3 [13].

6.2 Threats to Internal Validity

Looking at the results of the experiment, we can conclude that empirical evidence for the relationship between the independent and the dependent variables exists. We have tackled different aspects that could threaten the internal validity of the study:

- Differences among subjects. The error variance due to differences among students is reduced by using a within-subjects design.
- Learning effects. The counterbalancing procedure (subjects were randomly assigned in two groups) cancelled the learning effect due to similarities and the order of application of both techniques.
- Knowledge of the universe of discourse. We used the same case studies for all subjects.
- Fatigue effects. Each student took 45 minutes on average per session to apply both techniques. In this case, fatigue was not a relevant factor.
- Persistence effects. The students had never done similar experiments before.
- Subject motivation. The students were motivated because they had to apply FP in order to estimate required effort in their projects assigned for the semester.

6.3 Threats to External Validity

Two threats to external validity were identified which limited the ability to apply any such generalization:

- Materials. We used representative case studies in which students had to identify association, aggregation and composition relationships between classes. However, more empirical studies are needed that make use of software requirements specifications [10].
- Subjects. We are aware that more experiments with practitioners must be carried out in order to generalize these results. Although the graduate students

(practitioners) had experience in software development projects, they had not used FP technique in their projects before the experiment.

6 Conclusions and Future Work

This paper presented a conversion model to determine FP logic files using an analysis class diagram. This model considers the IFPUG CPM 4.2.1 rules and the composition relationship between classes that are not included in others approaches.

We also described two controlled experiments with undergraduate and graduate students in order to determine the accuracy of our approach compared to the IFPUG CPM 4.2.1 technique. The results of the experiments show that our approach produces at least equal results in accuracy when compared to the original IFPUG CPM 4.2.1 rules, and students (undergraduate and graduate) perceived that our approach is easier to use than the IFPUG FPA. Although the results obtained from the experiment are very encouraging, we are aware that more experimentation is needed to confirm them.

The future work regarding this research is:

- To conduct experiments with software requirements specification in order to get more results and opinions about the applicability of our model in the industry.
- To include generalization and association relations in our conversion model.
- To define rules to transform UML diagrams into IFPUG FPA transactions (EI; EO and EQ).

References

1. Abrahão, S., Poels, G., Pastor, O., Assessing the Reproducibility and Accuracy of Functional Size Measurement Methods through Experimentation, Proceedings ISESE 2004, IEEE Computer Society, 2004.
2. Abrahão, S., Poels, G., Experimental evaluation of an object-oriented function point measurement procedure, Information & Software Technology, Elsevier, 2007.
3. Albrecht, A. J., Measuring Application Development Productivity, in IBM Applications Development Symposium, Monterey, CA, 1979.
4. Basili, V. R., Caldiera, G. and Rombach, H. D., Goal Question Metric Paradigm, Encyclopedia of Software Engineering, ed. J. J. Marciniak, Wiley 1994.
5. Caldiera, G., Antoniol, G., Fiutem, R., Lokan, C., Definition and Experimental Evaluation of Function Points for Object-Oriented Systems, Proceedings METRICS'98, IEEE Computer Society, 1998.
6. Cantone, G., Pace, D., Calavaro, G., Applying Function Point to Unified Modeling Language: Conversion Model and Pilot Study, Proceedings of METRICS'04, IEEE Computer Society, 2004.
7. Carver, J., Jaccheri, L., Morasca, S., Issues in Using Students in Empirical Studies in Software Engineering Education, Proceedings METRICS'03, IEEE Computer Society, USA, 2003.
8. Coad, P., North,D., Mayfield,M., Object Models: Strategies, Patterns and Applications, Prentice-Hall, USA, 1997.
9. Fetcke, T., Abran, A. and Nguyen, T., Mapping the OOJacobson Approach into Function Point Analysis, IEEE Proceedings of TOOLS-23'97, 1997.

10. IEEE Computer Society, IEEE Std 830-1998, Recommended Practice for Software Requirements Specifications, The Institute of Electrical and Electronics Engineers, USA, 1998.
11. IFPUG, Function Points Counting Practices Manual (version 4.1.1), IFPUG: International Function Point User Group, Westerville Ohio, 2000.
12. IFPUG, Function Points Counting Practices Manual (version 4.2.1), IFPUG: International Function Point User Group, Westerville Ohio, 2004.
13. ISO. ISO/IEC 14143-3 - Information technology -- Software measurement -- Functional size measurement -- Part 3: Verification of functional size measurement methods, 2003.
14. Jaaksi, A., A Method for Your Object-Oriented Project, Journal of Object-Oriented Programming, Vol 10. No 9, 1998.
15. Jacobson, I., Object-Oriented Software Engineering. A Use Case Driven Approach, Addison-Wesley, USA, 1992.
16. Juristo, N., Moreno, A.M., Basics of Software Engineering Experimentation, Kluwer Academic Publishers, Boston, 2001.
17. Larman, C., Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative Development, Third Edition, Addison-Wesley, 2004
18. Pow-Sang, J.A., Imbert, R., Including the Composition Relationship among Classes to Improve Function Points Analysis, Proceeding VI Jornadas Peruanas de Computación-JPC'07, Trujillo, Peru, 2007.
19. Object Management Group, OMG Unified Modeling Language, http://www.uml.org, USA, 2005.
20. Uemura, T., Kusumoto, S. and Inoue, K., Function Point Measurement Tool for UML Design Specification. in 5th International Software Metrics Symposium (METRICS'99), Florida, USA, 1999, 62-69.