How Simple is it to Measure Software Size and Complexity for an IT Practitioner?

Robiolo, Gabriela

How Simple is it to Measure Software Size and Complexity for an IT Practitioner?

Gabriela Robiolo
Informatics Department
Austral University
Buenos Aires, Argentina
grobiolo@austral.edu.ar

Abstract— **An empirical study was conducted in order to evaluate the simplicity of FPA, COSMIC and Paths, from the IT practitioners' viewpoint. The results have shown that P are the simplest measure because they presented a significantly lower measurement time. The study has also been useful to see which aspects of Paths should be clarified in order to facilitate the practitioners' application of this measure and to demonstrate in a mathematical form how the complexity of an application which is measured with P increases if the number of "transactions" increases. The results obtained cannot be generalized because the number of measurements performed was small, and the subjects had particular characteristics, but a new empirical study using use case textual descriptions of different domains and subjects with different characteristics is expected to solve this limitation.**

Keywords; functional size measurement; functional size measurement simplicity; Function Points; COSMIC; Paths; empirical study.

I. INTRODUCTION

Size and complexity are important attributes which should be measured in order to appraise a developed software product, or one to be developed. Briand et al. [1] have defined software size according to the number of elements in a product, and software complexity according to the relationships among its elements, and they have highlighted the idea that they capture different software characteristics. However, although size and complexity can be identified as two different attributes, they are not completely independent: a big software product is always a complex product.

At present, when trying to measure size of software to be developed, the most widely used method is the Function Point (FP) counting manual, which was developed by the International Function Point User Group (IFPUG) [2], which adopted the original proposal made by Albrecht, that is to say, Function Point Analysis (FPA). FPA has the characteristics that it considers the product functionality viewed from the user's perspective, and that it can be used with any technology. Besides, another standard which is being increasingly used to measure the functional size of products is COSMIC function points [3].

Lavazza et al. have shown that the measurement of both FP and CFP can be based on suitable UML models [4][5][6][7]. For those organizations that adopt UML as the notation for requirements specification and design, measurement requires very little additional effort, as the functional size measure (FSM) is based on UML diagrams. Besides, it is more reliable, since there is little risk that the requirements specification artifacts will not be correctly understood. Quite interestingly, UML models that are suitable for supporting FSM include the description of use cases and scenarios, through use case diagrams and sequence diagrams.

Robiolo et al. have proposed to enhance the effort prediction capability of FSM by counting the "transactions" (T) and "paths" (P) that are described in use cases [8][9][10]. The concept of transaction described in [9] is similar to the concept of transaction in FPA, in that it concerns the interactions of the external users with the system, but it is substantially different from FPA, as transactions are identified at a finer grain (a FPA transaction can correspond to multiple T "transactions"). On the contrary, the measurement of paths is completely new: it addresses the complexity of use cases.

More recently, these authors [11] have shown that measurement-oriented UML modeling can support the measurement of both functional size and functional complexity. They have shown, by means of an empirical study, that it is reasonably easy to derive different types of FSM, as well as complexity measures, from UML models.

When the measure P was introduced by Robiolo et al., they did so in order to introduce a simpler measure. At that moment, that simplicity was accounted for by a theoretical explanation [12], but an empirical study was still pending. Another important concern was that this measure should be accepted by people from the software industry, such as developers, leaders, managers, and so on. In addition, it seemed that a more detailed description of "paths" identification had become necessary.

This paper describes an empirical study which evaluated how simple it is to measure with FPA, CFP and P, as well as how easy it is to understand the concept of such measures. To do so, the time used to measure the size and complexity of a software product was measured, and the dispersion of such measurements was calculated. The shorter the time, the simpler the measure would be, and the smaller the dispersion, the bigger the understanding.

The objects that were measured were the textual descriptions of the use cases of software products. The experiment subjects were IT practitioners who were not necessarily working on software development at that moment and, on top, some of them may even had little experience in this field.

Also, as the same objects were measured by different persons under similar conditions, the measurement differences were used to acknowledge those measurement aspects which should be further explained or clarified, especially when using P, as it is a relatively new measure.

The paper is organized as follows: section 2 accounts for related work; section 3 describes the empirical study; section 4 develops the data analysis and interpretation; while section 5 draws some conclusions.

II. Related Works

While searching in IEEE, ACM and Elsevier, only one paper that develops an empirical study which focuses on measurement time and dispersion when using FSM was found, although several presented empirical studies which dealt with the simplification of FSM.

Abrahao and Poels [13] presented an empirical study that evaluates OO-Method Function Points (OOmFP), a functional size measurement procedure for object-oriented systems that are specified by using the OO-Method approach. A laboratory experiment with undergraduate students was conducted to compare OOmFP with the FPA procedure on a range of variables, including efficiency, reproducibility, accuracy, perceived ease of use, perceived usefulness and intention to use. The results showed that OOmFP is more time-consuming than FPA, but the measurement results are more reproducible and accurate. The results also indicated that OOmFP is perceived to be more useful and more likely to be adopted in practice than FPA in the context of OO-Method systems development. The main differences with the empirical study here presented are: the subjects in the Abrahao and Poels study were undergraduate students with a similar educational background, the requirements documents used in such experiment followed the IEEE 830 Standard and also included an Entity-Relationship diagram and screen prototypes, the subjects were trained on each measure for two hours and they were given a more detailed written explanation on each measure.

In Symons's Mark II function points [14], a system consists of logical transaction-types, i.e., of logical input/process/output combinations. Entity-types are the equivalent of logical file types of FPA. The number of entity-types referenced by a transaction-type is considered its measure of complexity. The number of Mark II unadjusted FP is given by the weighted sum of input data element types, output data element types and entity-types references, where input and output data elements are the input and output operations of a transaction-type. The concept of transaction-types is similar to FPA, but the External Inquiries are considered just as any other input/process/output process combinations. This aspect is a convenient simplification, as in our empirical study the counting of Inquiries affects the dispersion of the measurement values. Moreover, if the Symons's Mark II function points method had been used, we would have expected to obtain a measurement time smaller than that obtained with FPA and COSMIC, and similar to that obtained with Paths.

FP have several Base Functional Components (BFC), which is a weakness, as the variability of function point counting is increased. Kemerer investigated this aspect and reported that the differences in function point measures of the same system yielded by different counters averaged 12.2% [15]. Jeffery et al. reported worse figures: a 30% variance within an organization, which rose to more than 30% across organizations [16]. Kitchenham and Kansala [17] expect that simpler counting would reduce the variability of the results, which has been confirmed by our empirical study.

Another aspect that contributed to the simplification of FP is the elimination of the weighted factors. They were critically investigated by Morasca [18], who analyzed the intuition behind weighted sums in the definition of measures and their theoretical and empirical validation pitfalls, which may make them questionable when used in the definition of meaningful and practically useful measures. Our paper casts further empirical evidence on this topic.

Conte et al. [19] defined the Early & Quick Function Points technique as a process which is capable of delivering measures quickly (i.e., without performing a full-fledged measurement process) or on the basis of incomplete information (e.g., when the knowledge of the user requirements is incomplete). The Early & Quick Function Points technique allows the user to count BFC that have not yet been properly classified (e.g., a transaction that is not yet known to be an output or a query) or even sets of BFC for which it is not yet possible to say how many transactions will be involved. All this has become possible due to the weights (computed via statistical analysis on the ISBSG dataset) which are attributed to aggregations of transactions or to transactions that have not yet been classified, or whose complexity is unknown. Early & Quick Function Points do not aim at simplifying the definition of Function Points (or CFP). Rather, they aim at providing the best approximation of the size measure in presence of limited information. This approach will not necessary reduce the measurement time.

Gencel and Demirors critically reviewed the FSM methods [20]. They stated that there are significant improvement opportunities for FSM, both theoretically and practically. They mentioned a few open issues, including the problem with the level of abstraction when decomposing Functional User Requirements into transactions, which affects the effort required to identify the transactions and the granularity level of measurement attributes. To avoid this in our empirical study, the level of abstraction of all the use case textual descriptions that were used was similar. Another aspect which was considered by Gencel and Demirors was the level of detail with which the Functional User Requirements should be defined in order to make a measurement reliable. These authors showed that it is the level of requirement detail what affects the measurement time of a software product. They suggested that both the existing and new methods should report the granularity level of their FSM process, so that the users of the methods will know when a specific method can be applied and how much effort will be required for its application. We believe that it would be possible to apply Paths to different documents which complement the information of the use case textual description, and that such deeper detail would improve the effort estimation error.

In fact, the best solution to improve the measuring time is to automate the process of measurement. There are several proposals to automate the FP and COSMIC computation [21][22]. More recently, Hussain et al. [23] have shown that an approximation of the COSMIC functional size can be automatically extracted from informally written textual requirements by using a supervised text mining approach. Such requirements are expressed as user stories. This is what happens in agile software development processes, in which requirements are written in a very sketchy way. Also, Arevalo et al. [24] are now working in order to automatically obtain the measure P from code.

III. Description of the Empirical Study

A. *Definition*

The empirical study was developed in the context of a continuing education program for IT practitioners from different educational and work backgrounds. The participants attended a workshop which had two objectives, one oriented to the subjects and another oriented to the development of the empirical study. The workshop gave the participants the opportunity to understand the concepts of size and complexity of a software product, to use such concepts in order to measure a particular product and to acknowledge the scope of these concepts, which they might use in future decision-making.

As regards the empirical study, the objective was to identify the simplest measure, considering that *a measure is simpler if it yields a smaller mean measurement time and if it displays a smaller standard deviation measurement value, and it is easier to understand if it displays a smaller normalized standard deviation measurement value.* This objective was achieved by comparing the measurement time obtained by using P and those times obtained by using the traditional measures FP and CFP. Besides, the dispersion obtained with each measure was calculated.

B. *Planning*

The *experimental subjects* were chosen from among the participants in a continuous education program for IT practitioners. They were expected to have experience with use cases and conceptual and logical data modeling. The participants were asked about the following aspects:

- Professional experience (they had to state the number of months they had worked in IT)
- Experience with FPA and COSMIC
- Use case experience (they had to inform the number of use cases that they had written/read (less than 20, 20-100, more than 100)
- Experience with Data Base development (number of months worked on this)

The *experimental objects* were the use case textual descriptions of two similar applications (P1 and P2) which used mobile phones and an internet interface. The first application was a system that allows the purchase of movie tickets through mobile phones, by a using prepaid card. At any time, and at any place, customers can access the application and purchase a ticket. The second application was a system that facilitates communication between long distance buses of a company and their headquarters, through mobile phones, thus enabling continuous communication and exchange of information. Its purpose was to monitor and control the flow of buses. “Fig. 1” shows the names of the use cases in each application.

Some aspects were controlled to facilitate the comparison:

- Use Case textual description size: the two projects had similar use case size, as the reading time may affect the results. The original textual descriptions were modified in order to avoid the use case level of detail problem [25] and to have a similar reading time.
- Explanation of each measure: The participants received an explanation which was similar in length and depth. The objective was to describe the principal aspects of each measure, focus on its elements, how they are identified, and how they are counted. “Fig. 2 and 3” show the slides used. Also, the excel sheets the participants had to work with were very clearly designed and they automatically performed the necessary calculations.

The workshop was run following these steps:

1) *The participants were divided in two groups.* The group members were distributed according to their previous experience with FP, CFP or P, so that well balanced groups were obtained. Each participant defined the sequence to measure the projects.

a) *Group A:* measured P1 with CFP and P2 with P.

b) *Group B:* measured with FP P1 and P2 with P.

2) *A set of materials was given to the participants,* which included: use case textual descriptions, the professor’s slides and two excel sheets -one for each measure- where they had to register the data to be collected. The sheets were designed to facilitate the measures understanding, and they automatically made all the calculations required.

3) *Each measure was explained to the participants*. The same amount of training on the different measures was provided. Also, an example of each was explained.

4) *The participants measured the applications,* and the measurement values they obtained were collected. Also, the time taken to perform the measurements was recorded.

5) *The data was collected and the results were analyzed with the participants.*

C. *Execution*

The characteristics of the participants are described in Table 1. The experience, ranged from less than one year to twenty-three-year-experience. Only one had previous experience with FMS and only two had significant experience with use cases. Almost all had data base experience, which facilitated the understanding of related concepts such entity, element data, query, etc.

Originally, the complete data set had 22 measurements, but only 17 of them were selected because some of the participants did not understand the core idea of some measures, so some measurements were incomplete.

The existence of minor omissions or common errors was not a condition to discard a measurement, because such omissions or common errors affected measurement results and, as the more complex a measure is, the more errors there will be, omissions and errors were part of our objects of study.

Table 2 shows the measurements that complied with the measurement requirements. Although the participants had to work with two use case textual descriptions, some did not have enough time to finish or did not comply with the measurement requirements.

Table 3 shows the results of the statistic analysis.

D. Threats to validity

The brief explanation given to the participants about the measures is the major weakness of this empirical study. It was not possible to give an extensive explanation of each measure, as there was not enough time (the whole workshop took 3 hours). However, this drawback may be transformed into a strength if we consider that in this empirical study the measurement time and the dispersion that novice measurers may work with has been measured. Also, the results have pointed out that there are measurement aspects which must be made clearer.

The two use case textual descriptions used in the empirical study came from a particular environment, so general conclusions cannot be reached because there may be different measurement problems in different environments. Bringing use cases from different types of environment for a future study would show if there are different behaviors.

Besides, as the use case detail level may influence the measurement results the detail levels of P1 and P2 were matched so that they became similar.

TABLE 1. WORKSHOP PARTICIPANTS

Id	Work Experience (Months)	Measuring Experience	Number of Use Cases Read or Written [<20, 20<CU<100, >100]	Data Base Experience (Months)
1	228	NO	<20	12
2	12	NO	<20	24
3	216	NO	<20	24
4	24	NO	<20	120
5	20	NO	<20	48
6	252	NO	>100	60
7[a]	-	-	-	-
8	91	YES	>100	60
9	3	NO	<20	45
10	282	NO	<20	180
11	192	NO	<20	36
12[a]	-	-	-	-

a.Two participants not completed personal data

TABLE 2. MEASUREMENT RESULTS

	FP		COSMIC		Paths	
Id	*FP*	*Time (minutes)*	*COSMIC*	*Time (minutes)*	*P*	*Time (minutes)*
1			134.00	49.00		
2					59.00	9.00
3			84.00	32.00	50.00	17.00
4			104.00	25.00	69.00	11.00
5					87.00	20.00
6			127.00	44.00		
7	177.00	28.00				
8	165.00	40.00			64.00	40.00
9	198.00	73.00			73.00	22.00
10	46.00	12.00			54.00	8.00
11	72.00	13.00				
12			108.00	33.00		

IV. DATA ANALYSIS AND INTERPRETATION

There were two aspects to be analyzed: measurement time and the dispersion presented by the measurement of size (FP and CFP) and complexity (P). Also, the different measurements made by novice measurers were expected to highlight measure aspects that should be made clearer. In particular, as P is quite a new measure, this first utilization of P by IT practitioners would lead to a more precise definition for practical counting.

TABLE 3. RESULTS OF THE STATISTIC ANALYSIS

	FP		COSMIC		Paths	
	FP	*Time*	*CFP*	*Time*	*P*	*Time*
Mean	131.60	33.20	111.40	36.60	65.14	18.14
Standard Deviation	67.94	25.07	19.82	9.71	12.56	11.07
Normalized Standard Deviation	0.52		0.18		0.19	

A. Measurement time

When analyzing the mean measurement time, we found out that P are the simplest, if they are compared to FP or CFP, because they resulted in a significantly lower mean beasurement time. However, although COSMIC has only one BFC, it did not seem to be simpler than FP because the measurement time obtained when measuring with CFP was similar to that obtained when measuring with FP. The four different sub-types of data movement in COSMIC and the identification of the data groups introduced a level of counting complexity similar to that presented by the five

BFC of FPA and their complexity classification. Moreover, as the multiplication of the FPA weighted factors was simplified by using an Excel sheet, the calculation of FP did not differ much from that of CFP.

FP presented a higher standard deviation measurement time, if the result is compared to that obtained with P and CFP, which pointed out that the identification and classification of the BFC may bring more difficulty to particular persons, so they may take longer to perform the task. Also, as the difference between the results obtained with P and CFP is not big, it may be concluded that they present a similar difficulty.

B. Size and complexity measurement dispersion

Another interesting aspect to be noted is the size and complexity measurement dispersion: the results pointed out that there may be a measurement subjective component, i.e., the different personal interpretations that may be made of a given text.

P and CFP presented similar normalized standard deviation values. However, FP presented a higher normalized standard deviation. In this empirical study, the main reason for this is the different interpretations that were made of the external inquiry. Some counted all the external inquiries included in each transactional function, and others considered them only as a transactional function.

A previous paper [13] has reported a FP standard deviation equal to 17.44. The big difference, if it is compared to the result obtained in this paper, points out that both a more intensive training and requirements with a higher level of detail using complementary documentation (i.e. Entity-Relationship diagram and screen prototypes) may reduce the standard deviation values. However, as the subjects of our experiment were less familiar with software development than the subjects of the previous study, a bigger standard deviation was acceptable.

It was also interesting to analyze the causes that led to accepting or discarding the measurements and the differences that the measurement presented. They can be summarized as follows:

- FPA: 5 measurements were made and they were all accepted. The differences between the 5 accepted measurements were that the participants applied a different criterion to identify the external inquiry or they thought of a different design.
- COSMIC: 7 measurements were made, 5 of which were accepted and 2 discarded. Those participants involved in the latter either repeated data groups or did not identify the data movement correctly. The differences in the 5 accepted measurements were that the participants applied a strict criterion not to count repeated things or they counted with an acceptable level of repetition (e.g. a query that is included in several functions) or they thought of a different design.
- Paths: 10 measurements were made, 7 of which were accepted and 3 discarded. Those whose measurements were discarded did not understand the concept of “transaction” or “path”, or made an incomplete measurement. The differences in the 7 accepted measurements were that the participants thought of a different design, they counted one actor stimulus per element data or they counted some alternative paths (AP) in a different manner.

The number of discarded measurements pointed out that 5 participants would have needed a more detailed explanation, or perhaps more time, to understand the principal concepts. This may be due to the big differences in work experience. The analysis of the discarded measurements also pointed out that the major problem when using CFP was the identification of the data groups or data movements, not the identification of each sub-type “entry”, “exit”, “read” or “write”. Paths proved to be not as easy as it seemed to be. The particular characteristic of a “transaction” having a finer granularity than a functional transaction, i.e., the fact that the dialogue between the actor and the system increases the number of “transactions” in a use case, seemed to be an obstacle. Moreover, the differences found in the 8 correct measurements of Paths have been used to clarify some aspects of P counting.

For most of the practitioners, this was their first size or complexity measurement of a software product. The workshop was an opportunity for them to understand that the size and complexity of a product may be measured. They could also understand how the measurement of software size and complexity may impact some of the everyday decisions they have to make at work.

C. Paths aspects that affect the measurements

P is the measure that measures the complexity of a “transaction”, therefore, the first step is to identify a “transaction”, i.e., its beginning and ending. The beginning is the actor stimulus and the end is the reply that the actor receives from the system. Within each “transaction”, the main path (MP) and the AP -pointed out by the IF …THEN expression or similar textual expressions that have the same signification- have to be identified. Usually, in a textual description of use cases, the MP corresponds to the “transaction” sequence of steps in normal conditions and the AP to those under exceptional or particular conditions.

Also, each use case may be represented as a graph, in a similar form to that in which MacCabe [26] represented the code (see a detailed example in [11]). However, the complexity of an application is defined as the sum of the complexity of each component as it is shown in (1):

$$C(UC_1) + \ldots + C(UC_n) = C(Application). \qquad (1)$$

where UC_i are the use cases that make up the application and n is the number of use cases. Also, as each use case can be considered a set of paths (one MP and several AP), the complexity of an application is expressed as (2)

$$MP_1 + \sum AP_{1j} + \ldots + MP_n + \sum AP_{nk} = C(Application). \quad (2)$$

where j and k are the alternative paths of each *UC*. As the complexity of every MP equals 1, the complexity of an application is defined as shown in (3)

$$NAT + NAAP = C(Application). \quad (3)$$

where *NAT* is the number of application "transactions" and *NAAP* is the number of application AP, which shows that the complexity of an application increases if the number of "transactions" or number of AP increases.

Other aspects to be considered when "transactions" are counted are:

- The selection of a menu item is not counted as a "transaction".
- The response of the machine to a stimulus is part of the "transaction". "Transactions" begin with the stimulus and end with a status message or when the control is returned to the actor.
- Each "transaction" is identified when the actor triggers a button (e.g. "save", "remove", "run", etc), or when a question is answered (e.g. "The personal data will be removed. Do you want to remove the personal data? YES/NO").
- The use cases may capture an intensive dialogue between the actor and the machine. Each time the actor interacts with the machine in order to begin an execution, or to answer a question, a new "transaction" is identified. This aspect defines a different level of granularity in each use case, which is one of the main differences with FPA.
- Data elements do not affect the number of "transactions". Usually, each "transaction" has a packed set of data elements, which could have one data element, several data elements or data groups.
- A "transaction" should not be counted twice, although the wording in the text may be repeated to ensure comprehension of the text.
- A "transaction" does not have to be identified with a corresponding use case step, as it is a function. It may be a use case step, but in this case, it is a particular case of "transaction" definition.
- The selection of a data element from a set of pre-defined data is not considered an actor stimulus.
- Neither the acceptance nor the rejection of an error message is counted, as it does not define a new functionality, it only reports the state of the system or the "transaction", and it defines the end of a "transaction".
- The cancelation of a transaction is not counted.

Moreover, the aspects to be considered when the "paths" are counted are:

- The AP are not the number of steps of a use case.
- The error messages do not define AP. The AP are the different options offered in the text. When these different options are present in the code, they are within the IF THEN expressions.

V. CONCLUSIONS

This empirical study was conducted in order to assess the simplicity of FPA, COSMIC and Paths, from the IT practitioners' point of view. Our aim was to identify the simplest measure, taking into account that the shorter the time taken to measure a given product, the simpler the measure would be, and that the smaller the measurement dispersion, the higher the degree of understanding of the measure would be.

The results have shown that P are the simplest measure when compared to FP and CFP because they presented a significantly lower measurement time. Contrary to what had been thought when considering the number of BFC, and according to the results obtained in our empirical study, CFP should not be considered simpler than FP, since measuring with CFP took almost the same time as measuring with FP.

In addition, the study has contributed to pinpoint which aspects of Paths should be clarified in order to facilitate the practitioners' application of this measure. Moreover, we have demonstrated in a mathematical form how the complexity of an application which is measured with P increases if the number of "transactions" increases.

It is important to note that this empirical study has shown the difficulty that novice measurers may experience when working with these measures. However, this difficulty is not expected to come up when experienced measurers with strong skills in software development perform the measurements. Nonetheless, this experience has helped these IT practitioners (who are or may become managers some day) to understand the importance of measuring, as well as the need for training those who are in charge of measuring. Moreover, they have participated in an experiment that they can reproduce in the companies where they work in order to verify the errors they are working with.

This study has led us to conclude that the best solution to solve measurement deviation is the automatization of the measurement process.

Unfortunately, it is not possible to generalize the results obtained, as the number of measurements performed was small, and the subjects had particular characteristics. However, a new empirical study using use case textual descriptions of different domains and subjects with different characteristics would solve this limitation.

ACKNOWLEDGMENT

Our thanks to the Research Fund of Austral University, which made this study possible and to Luigi Lavazza for his opportune comments.

REFERENCES

[1] L.C. Briand, S. Morasca, V.R, Basili, "Property-Based Software Engineering Measurement," IEEE Transactions on Software Engineering, Vol. 22, 1996, pp. 68 – 86.

[2] ISO/IEC 20926: 2003, Software engineering – IFPUG 4.1Unadjusted functional size measurement method – Counting Practices Manual, International Organization for Standardization, Geneva.

[3] ISO/IEC19761:2003, Software Engineering – COSMICFFP– A Functional Size Measurement Method, ISO.

[4] L.Lavazza, V del Bianco and C. Garavaglia, "Model-based Functional Size Measurement," in Proceedings of 2nd International Symposium on Empirical Software Engineering and Measurement (Incorporating ISESE and Metrics, Kaiserslautern, Germany), ESEM 2008.

[5] L. Lavazza and C. Garavaglia, "Using Function Point in the Estimation of Real-Time Software: an Experience," Software Measurement European Forum (Milano). SMEF 2008.

[6] L. Lavazza, and C. Garavaglia, "Using Function Points to Measure and Estimate Real-Time and Embedded Software: Experiences and Guidelines," in Proceedings of 3rd Int. Symp. on Empirical SW Engineering and Measurement (October 15-16, Lake Buena Vista, Florida). ESEM 2009.

[7] L. Lavazza and V. del Bianco, "A Empirical study in COSMIC Functional Size Measurement: the Rice Cooker Revisited", IWSM/Mensura 2009 (4-6 November, Amsterdam).

[8] G. Robiolo, "A Simple Approach that Improves Early Effort Estimation Based on Use Cases," in Proceedings of the Third International Doctoral Symposium on Empirical Software Engineering (Kaiserslautern, Germany), IDoESE 2008.

[9] G. Robiolo, C. Badano, R. Orosco, "Transactions and Paths: two use case based metrics which improve the early effort estimation," In Proceedings of 3rd Int. Symp. on Empirical SW Engineering and Measurement (October S. 15-16, , Lake Buena Vista, Florida). ESEM 2009.

[10] G. Robiolo, and R. Orosco, "Employing use cases to early estimate effort with simpler metrics," Innovations Syst. Softw. Eng, vol.4, 2008.

[11] Luigi Lavazza and Gabriela Robiolo. 2010. Introducing the evaluation of complexity in functional size measurement: a UML-based approach. In Proceedings of the 2010 ACM-IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM '10). ACM, New York, NY, USA, , Article 25 , 9 pages.

[12] G. Robiolo, "A Simple Approach that Improves Early Effort Estimation Based on Use Cases. In Proceedings of the Third International Doctoral Symposium on Empirical Software Engineering (Kaiserslautern, Germany), IDoESE , 2008.

[13] S. Abrahao and G. Poels, "Experimental evaluation of an object-oriented function point measurement procedure", Information and Software Technology, 49 (4), 2007, pp. 366-380.

[14] C. Symons, "Function point analysis: Difficulties and improvements," IEEE Transactions on Software Engineering, 14:2–11, 1988.

[15] C. Kemerer, "Reliability of Function Points Measurement: A field experiment," Communications of the ACM, 1993

[16] J. Jeffery, G. Low, and M. Barnes, "Comparison of Function Point Counting Techniques," IEEE Transactions on Software Engineering, 19(5), 1993, pp. 529–532.

[17] B. Kitchenham and K. Känsälä, "Inter-Item Correlations among Function points,"International Conference on Software Engineering Proceedings of the 15th international conference on Software Engineering (Baltimore, Maryland, United States), 1993, pp. 477 – 480.

[18] S. Morasca, "On the use of weighted sums in the definition of measures," International Conference on Software Engineering. In 2010 ICSE: Workshop on Emerging Trends in Software Metrics, pages 8–15,2010.

[19] M. Conte, T. Iorio, R. Meli, and L. Santillo. "E&Q: An Early & Quick Approach to Functional Size Measurement Methods", in Software Measurement European Forum, 2004.

[20] C. Gencel and O. Demirors, "Functional Size Measurement Revisited," ACM Transactions on Software Engineering and Methodology, 17(3), 2008.

[21] T.Uemura, S. Kusumoto, and K. Inoue, "Function points measurement tool for UML design specification," inProceeding. of the Sixth International Software Metrics Symposium (Boca Raton, Florida, US), 1999.

[22] H.,Diab, F.Koukane, M.Frappier, R. St-Denis, "μcROSE: Automated measurement of COSMIC-FFP for Rational Rose RealTime," Information and Software Technology, 47 (3), 2005, pp. 151-166

[23] I. Hussain, L. Kossein, and O. Ormandjeva, "TowardsApproximating COSMIC Functional Size from User Requirements in Agile Development Processes Using Text Mining,"Natural Language Processing and Information Systems, Lecture Notes in ComputerScience, 6177/2010, 2010.

[24] G. Arevalo and G. Robiolo, and M. Martinez Soler, "Traceable complexity metric from requirements to code," in Proceedings of the 2010 ACM-IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM '10). ACM, New York, NY, USA, , Article 64 , 1 pages.

[25] K. Vinsen, D. Jamieson and G. Callender, "Use Case Estimation - The Devil is in the Detail," Proceedings of the Requirements Engineering Conference, 12th IEEE International RE'04, Vol. 00, 2004, pp.10-15.

[26] McCabe, T. 1976. A Complexity measure, IEEE Transactions on software Engineering, Vol. SE-2, NO. 4.

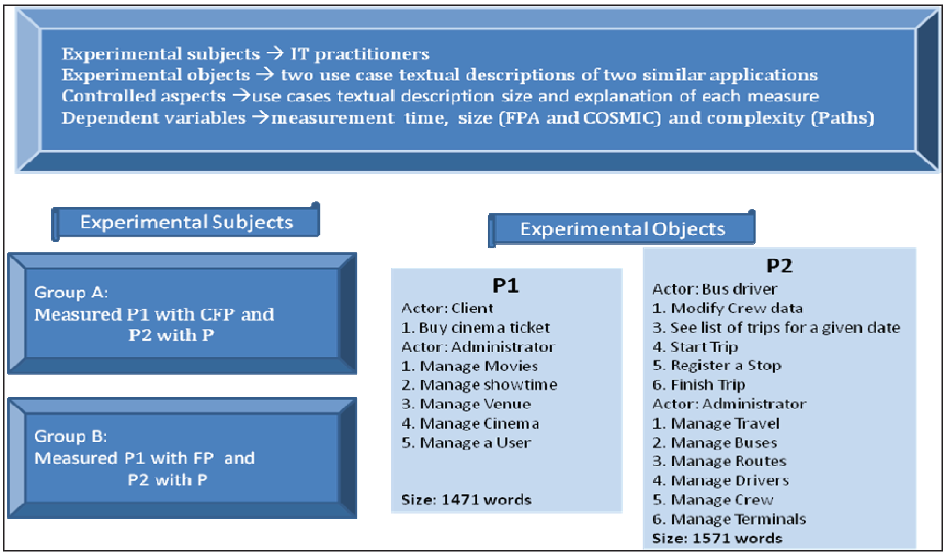

Figure 1. Empirical study summary.

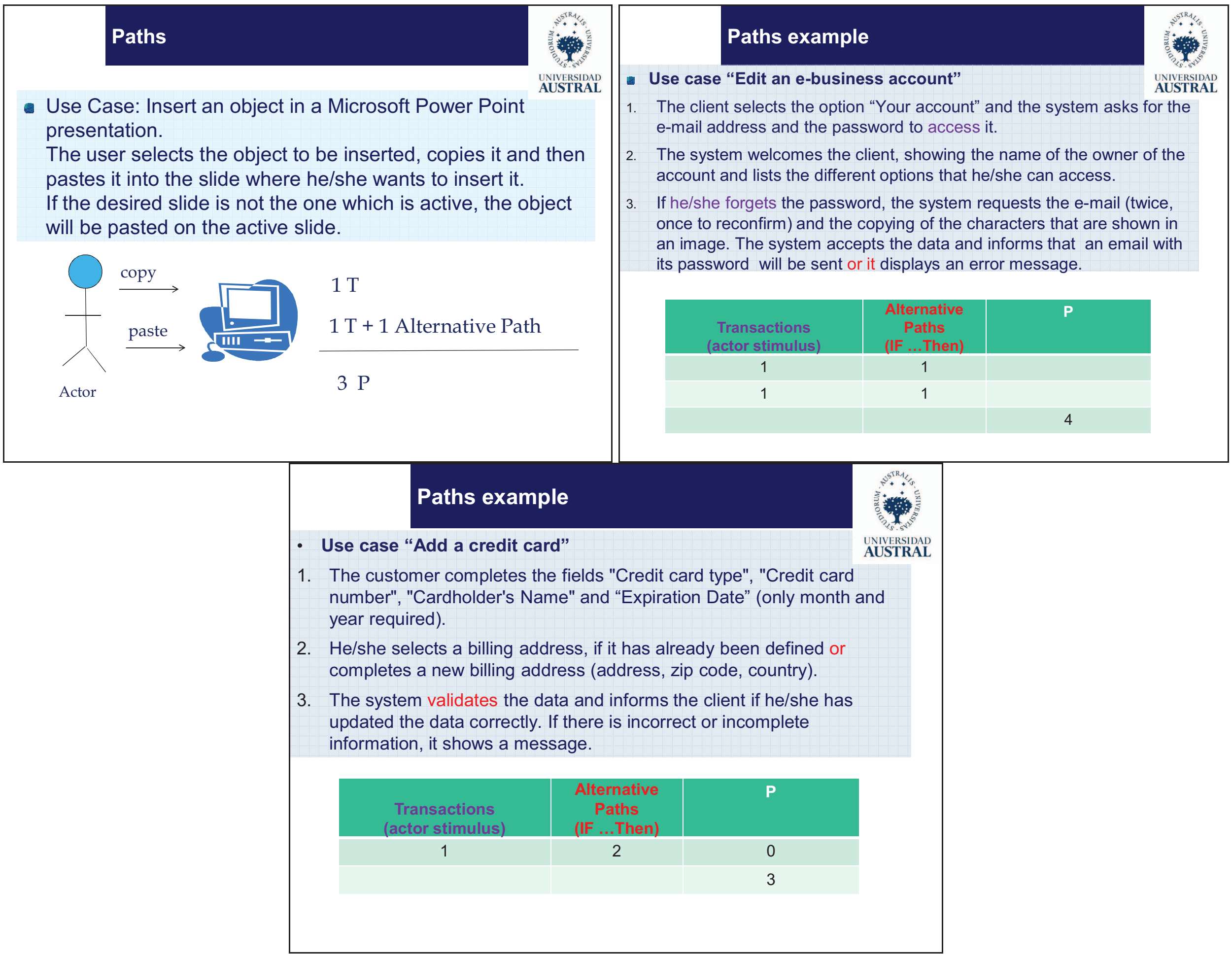

Figure 2. Example of the slides used in the wokshop

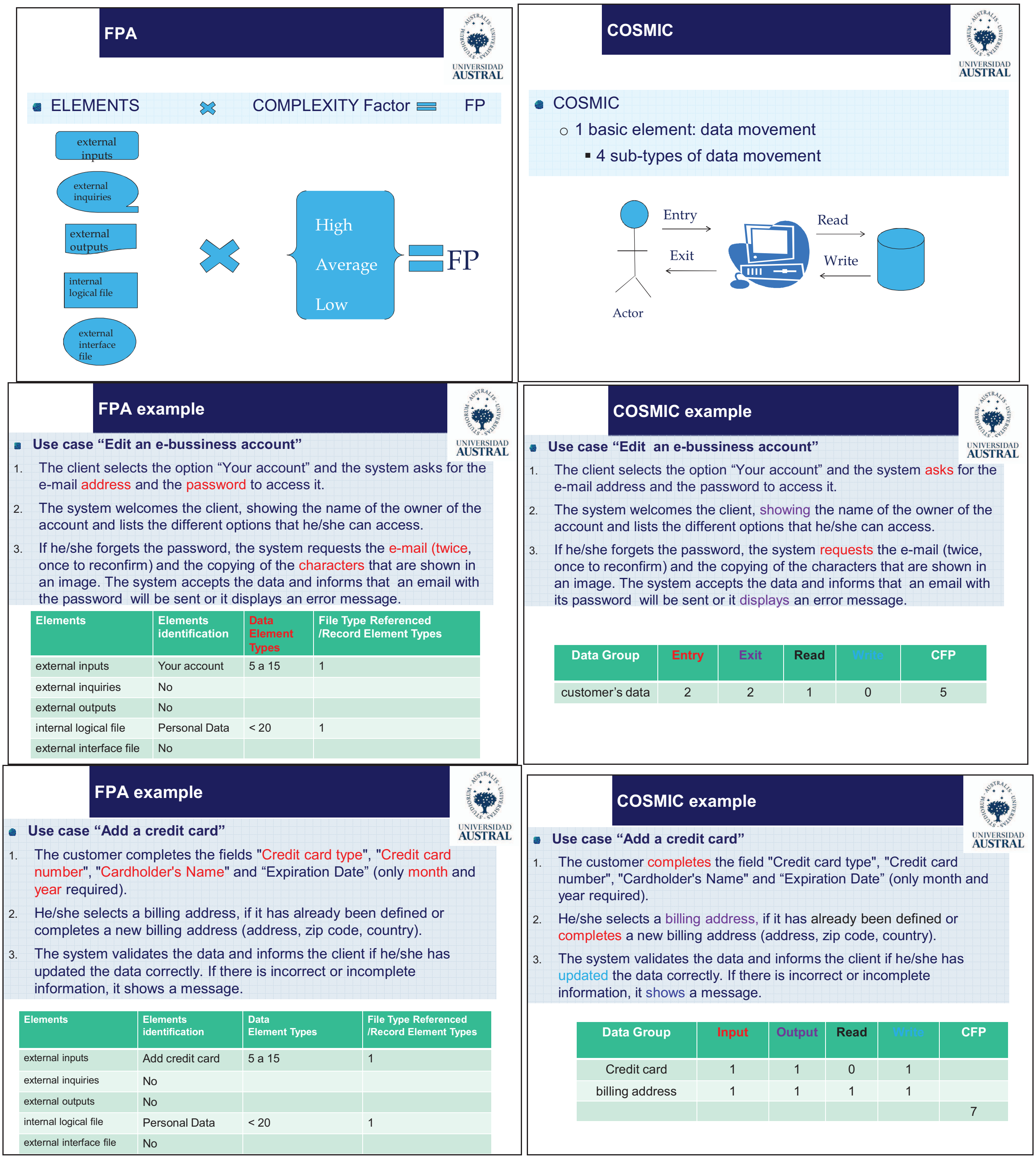

Figure 3. Example of the slides used in the wokshop