A Report on Using Simplified Function Point Measurement Processes

Luigi Lavazza   Geng Liu
Dipartimento di Scienze Teoriche e Applicate
Università degli Studi dell'Insubria
Varese, Italy
luigi.lavazza@uninsubria.it, giulio.liu@gmail.com

Abstract—**Background: Function Point Analysis is widely used, especially to quantify the size of applications in the early stages of development, when effort estimates are needed. However, the measurement process is often too long or too expensive or requires more knowledge than available when development effort estimates are due. To overcome these problems, simplified methods have been proposed to measure Function Points.**
Objectives: The work reported here concerns the experimentation of simplified functional size measurement methods in the sizing of both "traditional" and real-time applications. The goal is to evaluate the accuracy of the sizing with respect to full-fledged Function Point Analysis.
Method: A set of projects, which had already been measured by means of Function Point Analysis, have been measured using the NESMA and Early&Quick Function Points simplified processes: the resulting size measures were compared.
Results: while NESMA indicative method appears to quite overestimate the size of the considered applications, the other methods provide much more accurate estimates of functional size. EQFP methods proved more accurate in estimating the size of non Real-Time applications, while the NESMA estimated method proved fairly good in estimating both Real-Time and non Real-Time applications.
Conclusions: The results of the experiment reported here show that in general it is possible to size software via simplified measurement processes with an acceptable accuracy. In particular, the simplification of the measurement process allows the measurer to skip the function weighting phases, which are usually expensive, since they require a thorough analysis of the internals of both data and operations.

Keywords-Functional Size Measures; Function Points; Simplified measurement processes; Early&Quick Function Points (EQFP); NESMA estimated; NESMA indicative.

I. INTRODUCTION

Function Point Analysis (FPA) [1][4][2][3] is widely used. Among the reasons for the success of FPA is the fact that it can provide measures of size in the early stages of software development, when they are most needed for cost estimation.

However, FPA performed by a certified function point consultant proceeds at a rather slow pace: between 400 and 600 function points (FP) per day, according to Capers Jones [13], between 200 and 300 function points per day according to experts from Total Metrics [14]. As a consequence, measuring the size of a moderately large application can take too long if cost estimation is needed urgently. Also the cost of measurement can be considered excessive by software developers. In addition, cost estimates may be needed when requirements have not yet been specified in detail and completely. To overcome these problems, simplified FP measurement processes have been proposed. Among these are the NESMA (Netherland Software Metrics Association) indicative and estimated methods, and the Early & Quick Function Points method. The proposers of these methods claim that they allow measurers to compute good approximations of functional size measures with little effort and in a fairly short time.

The goal of the work reported here is to test the mentioned simplified functional size measurement processes on real projects in both the "traditional" and real-time domains. Function Points are often reported as not suited for measuring the functional size of embedded applications. The motivation is that FP –conceived by Albrecht when the programs to be sized were mostly Electronic Data Processing applications– capture well the functional size of data storage and movement operations, but are ill-suited for representing the complexity of control and elaboration that are typical of embedded and real-time software. However, a careful interpretation of FP counting rules makes it possible to apply FPA to embedded software as well [10].

In this paper we apply the International Function Points User Group (IFPUG) measurement rules [2] to size a set of programs for non-real time playing on the internet, and we apply the guidelines given in [8] (which are as IFPUG-compliant as possible) to measure a set of embedded real-time avionic applications. All these measures are used to test the accuracy of simplified functional size measurement processes.

The goal of the paper is therefore to evaluate if simplified functional size measurement processes can be used to size real-time and embedded applications, as well as "traditional" business applications.

The paper is organized as follows: Section II briefly introduces the simplified functional size measurement processes used in the paper. Section III describes the projects being measured and gives their sizes measured according to the full-fledged, canonical FPA process. Section IV illustrates the sizes obtained via simplified functional size measurement processes. Section V discusses the accuracy of the measures obtained via the simplified methods used and outlines the lessons that can be learned from the reported experiment.

Section VI accounts for related work. Finally, Section VII draws some conclusions and outlines future work.

II. A Brief Introduction to Simplified Size Measurement Processes

The FP measurement process involves (among others) the following activities:

– Identifying logic data;
– Identifying elementary processes;
– Classifying logic data as internal logic files (ILF) or external interface files (EIF);
– Classifying elementary processes as external inputs (EI), outputs (EO), or queries (EQ);
– Weighting data functions;
– Weighting transaction functions.

Simplified measurement processes allow measurers to skip –possibly in part– one or more of the aforementioned activities, thus making the measurement process faster and cheaper.

The most well-known approach for simplifying the process of FP counting is probably the Early & Quick Function Points (EQFP) method [5][7]. EQFP descends from the consideration that estimates are sometimes needed before the analysis of requirements is completed, when the information on the software to be measured is incomplete or not sufficiently detailed.

Since several details for performing a correct measurement following the rules of the FP manual [2] are not used in EQFP, the result is a less precise measure. The trade-off between reduced measurement time and costs is also a reason for adopting the EQFP method even when full specifications are available, but there is the need for completing the measurement in a short time, or at a lower cost. An advantage of the method is that different parts of the system can be measured at different detail levels: for instance, a part of the system can be measured following the IFPUG manual rules [2][3], while other parts can be measured on the basis of coarser-grained information. In fact, the EQFP method is based on the classification of the processes and data of an application according to a hierarchy (see Figure 1. [7]).

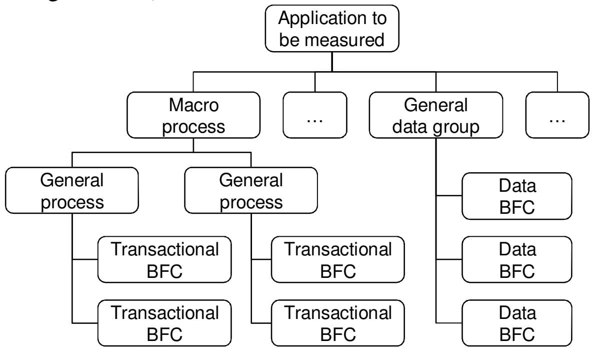

Figure 1. Functional hierarchy in the Early & Quick FP technique

Transactional BFC (Base Functional Components) and Data BFC correspond to IFPUG's elementary processes and LogicData, while the other elements are aggregations of processes or data groups. The idea is that if you have enough information at the most detailed level you count FP according to IFPUG rules; otherwise, you can estimate the size of larger elements (e.g., General or Macro processes) either on the basis of analogy (e.g., a given General process is "similar" to a known one) or according to the structured aggregation (e.g., a General process is composed of 3 Transactional BFC). Therefore, by considering elements that are coarser-grained than the FPA BFC, the EQFP measurement process leads to an approximate measure of size in IFPUG FP.

Tables taking into account the previous experiences with the usage of EQFP are provided to facilitate the task of assigning a minimum, maximum and most likely quantitative size to each component. For instance, TABLE I. provides minimum, maximum and most likely values for generic (i.e., not weighted) functions as given in [7]. Using this method involves the activities indicated in TABLE III. The time and effort required by the weighting phases are saved. Such saving can be relevant, since weighting a data or transaction function requires analyzing it in detail.

TABLE I. EQFP: FUNCTION TYPE WEIGHTS FOR GENERIC FUNCTIONS

Function type	**Complexity**		
	Low	***Likely***	***High***
Generic ILF	7.4	7.7	8.1
Generic EIF	5.2	5.4	5.7
Generic EI	4	4.2	4.4
Generic EO	4.9	5.2	5.4
Generic EQ	3.7	3.9	4.1

The size of unspecified generic processes (i.e., transactions that have not been yet classified as inputs, outputs or queries) and unspecified generic data groups (i.e., logical files that have not been yet classified as ILF or EIF) as given in [7] are illustrated in TABLE II. When using this method, only the identification of logical data and elementary processes needs to be done, as shown in TABLE III. Both the classification of data and transaction functions and their weighting are skipped. Consequently, sizing based on unspecified generic processes and data groups is even more convenient –in terms of time and effort spent– than sizing based on generic (i.e., non weighted) functions.

TABLE II. EQFP: FUNCTION TYPE WEIGHTS FOR UNSPECIFIED GENERIC PROCESSES AND DATA GROUPS

Function type	**Complexity**		
	Low	***Likely***	***High***
Unspefied Generic Processes	4.3	4.6	4.8
Unspefied Generic Data Group	6.4	7.0	7.8

Methods for simplifying the counting of FP, the Indicative NESMA method [6] simplifies the process by only requiring the identification of LogicData from a data model. The Function Point size is then computed by applying predefined weights, whose value depends on whether the data model is normalized in 3rd normal form:

Non normalized model: Function Points = Number of ILF × 35 + Number of EIF × 15

Normalized model: Function Points = Number of ILF × 25 + Number of EIF × 10

The process of applying the NESMA indicative method involves only identifying logic data and classifying logic data as ILF or EIF. Accordingly, it requires less time and effort than the EQFP methods described above, in general. However, it is quite clear that the Indicative NESMA method is quite rough in its computation. The official NESMA counting manual specifies that errors in functional size with this approach can be up to 50%.

The Estimated NESMA method requires the identification and classification of all data and transaction functions, but does not require the assessment of the complexity of each function: Data Functions (ILF and EIF) are assumed to be of low complexity, while Transactions Functions (EI, EQ and EO) are assumed to be of average complexity. Accordingly, the Estimated NESMA method is expected to be more approximated than the EQFP method based on generic functions, as the latter uses *likely* values for transactions of unknown complexity, derived from statistic analysis.

TABLE III. ACTIVITIES REQUIRED BY DIFFERENT SIMPLIFIED MEASUREMENT PROCESSES

Measurement activities	IFPUG	NESMA indic.	NESMA estim.	EQFP Generic func.	EQFP Unspec. generic func.
Identifying logic data	✓	✓	✓	✓	✓
Identifying elementary processes	✓		✓	✓	✓
Classifying logic data as ILF or EIF	✓	✓	✓	✓	
Classifying elementary processes as EI, EO, or EQ	✓		✓	✓	
Weighting data functions	✓				
Weighting transaction functions	✓				

The activities required by the simplified functional size measurement methods considered in the paper are reported in TABLE III. Of course, the IFPUG method requires all the activities listed in TABLE III.

III. THE CASE STUDY

A. *Real-time projects*

The real-time projects measured are from a European organization that develops software for avionic applications, and for other types of embedded and real-time applications.

The projects' FUR were modeled using UML as described in [9], and were then measured according to IFPUG measurement rules [2]. When the real-time nature of the software made IFPUG guidelines inapplicable, we adopted ad-hoc counting criteria, using common sense and striving to preserve the principles of Function Point Analysis, as described in [10].

The same projects were then sized using the NESMA and EQFP simplified functional size measurement processes, using the data that were already available as a result of the IFPUG measurement.

All the measured projects concerned typical real-time applications for avionics or electro-optical projects, and involved algorithms, interface management, process control and graphical visualization.

For each project the measurement of the functional size was carried out in two steps. First, a model of the product was built. The models were written in UML and represented the requirements, including all the information needed for the measurement of FPs and excluding the unnecessary details [9]. Then, the function points were counted, on the basis of the model, according to IFPUG rules.

TABLE IV. reports the size in FP of the measured projects, together with the BFC and –in parentheses– the number of unweighted BFC. For instance, project 1 involved 18 Internal Logic Files, having a size of 164 FP.

TABLE IV. REAL-TIME PROJECTS' SIZES (IFPUG METHOD)

Project ID.	ILF	EIF	EI	EO	EQ	FP
1	164 (18)	5 (1)	90 (21)	8 (2)	22 (5)	289
2	56 (8)	0 (0)	21 (6)	18 (3)	6 (1)	101
3	73 (7)	0 (0)	12 (2)	47 (8)	4 (1)	136
4	130 (15)	15 (3)	44 (11)	0 (0)	6 (1)	195
5	39 (4)	0 (0)	28 (8)	39 (8)	0 (0)	106
6	71 (9)	5 (1)	8 (2)	139 (28)	0 (0)	223
7	7 (1)	0 (0)	3 (1)	5 (1)	0 (0)	15

B. *Non Real-time projects*

The non real-time project considered are programs that allow users to play board or card games vs. remote players via the internet.

The projects were measured –as the real-time ones– in two steps: the UML model of each product was built along the guidelines described in [9]; then, the function points were counted, on the basis of the model, according to IFPUG rules.

TABLE V. reports the size in FP of the measured projects, together with the BFC and –in parentheses– the number of unweighted BFC.

TABLE V. NON REAL-TIME PROJECTS' SIZES (IFPUG METHOD)

Project ID.	ILF	EIF	EI	EO	EQ	FP
1	45 (6)	7 (1)	34 (10)	6 (1)	0 (0)	92
2	28 (4)	20 (4)	37 (9)	5 (1)	4 (1)	94
3	21 (3)	5 (1)	27 (7)	8 (2)	18 (6)	79
4	31 (4)	0 (0)	49 (16)	13 (3)	3 (1)	96
5	24 (3)	0 (0)	45 (14)	21 (5)	0 (0)	90
6	49 (7)	0 (0)	36 (9)	0 (0)	6 (2)	91

IV. RESULTS OF SIMPLIFIED MEASUREMENT

Simplified measurement processes were applied following their definitions, which require data that can be easily derived from the tables above. So, for instance, the data required for Real-Time project 1 are the following:

- The NESMA indicative method requires the numbers of ILF and EIF. TABLE I. shows that the number of ILF is 18, and the number of EIF is 1.
- The NESMA estimated method and the EQFP generic functions methods require the numbers of ILF, EIF, EI, EO and EQ. TABLE I. shows that the numbers of ILF, EIF, EI, EO and EQ are, respectively, 18, 1, 21, 2, and 5.
- The EQFP unspecified generic functions method requires the numbers of data groups (that is, the number of ILF plus the number of EIF) and the number of transactions (that is, the sum of the numbers of EI, EO and EQ). TABLE I. shows that the number of data groups is 18+1 = 19, and the number of transactions is 21+2+5 = 28.

A. Applying NESMA indicative

The applications to be measured were modeled according to the guidelines described in [9]. The logic data files –modeled as UML classes– provide a data model that cannot be easily recognized as normalized or not normalized. Therefore, we applied both the formulae for the normalized and not normalized models.

TABLE VI. MEASURES OF REAL-TIME PROJECTS OBTAINED VIA THE NESMA METHODS

Project ID	IFPUG	NESMA indicative non normalized	NESMA indicative normalized	NESMA estimated
1	289	645	460	245
2	101	280	200	99
3	136	245	175	101
4	195	570	405	168
5	106	140	100	100
6	223	330	235	216
7	15	35	25	16

TABLE VII. MEASURES OF NON REAL-TIME PROJECTS OBTAINED VIA THE NESMA METHODS

Project ID	IFPUG	NESMA indicative non normalized	NESMA indicative normalized	NESMA estimated
1	92	225	160	81
2	94	200	140	82
3	79	120	85	73
4	96	140	100	91
5	90	105	75	83
6	91	245	175	82

The formulae of the NESMA indicative method were applied to the number of ILF and EIF that had been identified during the IFPUG function point counting process. The results are given in TABLE VI. for Real-Time projects and in TABLE VII. for non Real-Time projects.

B. Applying NESMA estimated

The formulae of the NESMA indicative method were applied to the number of ILF, EIF, EI, EO, and EQ that had been identified during the IFPUG function point counting process. The results are given in TABLE VI. for Real-Time projects and in TABLE VII. for non Real-Time projects.

C. Applying EQFP

As described in Figure 1. , the EQFP method can be applied at different levels. Since we had the necessary data, we used the BFC aggregation level.

TABLE VIII. MEASURES OF REAL-TIME PROJECTS OBTAINED VIA THE EQFP METHOD

Project ID	IFPUG	EQFP – unspecified generic processes and data groups	EQFP –generic transactions and data files
1	289	262	262
2	101	102	106
3	136	100	108
4	195	181	182
5	106	102	106
6	223	208	229
7	15	16	17

At this level it is possible to use the data functions and transaction functions without weighting them or even without classifying transactions into EI, EO and EQ and logic data into ILF and EIF. In the former case (generic functions) the weights given in TABLE I. are used, while in the latter case (unspecified generic functions) the weights given in 0are used.

The results of the application of EQFP are given in TABLE VIII. for Real-Time projects, and in TABLE IX. for non Real-Time projects.

TABLE IX. MEASURES OF NON REAL-TIME PROJECTS OBTAINED VIA THE EQFP METHOD

Project ID	IFPUG	EQFP – unspecified generic processes and data groups	EQFP –generic transactions and data files
1	92	100	99
2	94	107	99
3	79	97	92
4	96	120	118
5	90	108	108
6	91	100	100

V. SUMMARY AND LESSONS LEARNED

To ease comparisons, all the size measures of RT projects are reported in TABLE X. and those of non RT projects are reported in TABLE XI.

TABLE X. MEASURES OF REAL-TIME PROJECTS OBTAINED VIA THE VARIOUS METHODS

Proj ID	IFPUG	NESMA ind. non norm.	NESMA ind. norm.	NESMA estim.	EQFP unspec.	EQFP generic
1	289	645	460	245	262	262
2	101	280	200	99	102	106
3	136	245	175	101	100	108
4	195	570	405	168	181	182
5	106	140	100	100	102	106
6	223	330	235	216	208	229
7	15	35	25	16	16	17

TABLE XI. MEASURES OF NON REAL-TIME PROJECTS OBTAINED VIA THE VARIOUS METHODS

Proj ID	IFPUG	NESMA ind. non norm.	NESMA ind. norm.	NESMA estim.	EQFP unspec.	EQFP generic
1	92	225	160	81	100	99
2	94	200	140	82	107	99
3	79	120	85	73	97	92
4	96	140	100	91	120	118
5	90	105	75	83	108	108
6	91	245	175	82	100	100

It is easy to see that the NESMA indicative method yields the greatest errors. On the contrary, the NESMA estimated and EQFP methods yield size estimates that are close to the actual size.

The relative measurement errors are given in TABLE XII. and TABLE XIII. , where the least error for each project is in bold. It is easy to see that the NESMA indicative methods are generally outperformed by the other methods. For Real-Time projects EQFP (either in the unspecified or generic flavor) tend to provide the most accurate results, while the NESMA estimated method provides quite reasonable estimates. For non Real-Time projects the NESMA estimated method appears even better than the EQFP methods. Quite noticeably, NESMA estimated underestimates all non Real-Time projects except one.

TABLE XII. RELATIVE MEASUREMENT ERRORS (REAL-TIME PROJECTS)

Proj ID	NESMA ind. non norm.	NESMA ind. norm.	NESMA estim.	EQFP unspec.	EQFP generic
1	123%	59%	-15%	**-9%**	**-9%**
2	177%	98%	-2%	**1%**	5%
3	80%	29%	-26%	-26%	**-21%**
4	192%	108%	-14%	**-7%**	**-7%**
5	32%	-6%	-6%	-4%	**0%**
6	48%	5%	**-3%**	-7%	**3%**
7	133%	67%	**7%**	**7%**	13%

TABLE XIII. RELATIVE MEASUREMENT ERRORS (NON REAL-TIME PROJECTS)

Proj ID	NESMA ind. non norm.	NESMA ind. norm.	NESMA estim.	EQFP unspec.	EQFP generic
1	145%	74%	-12%	9%	**8%**
2	113%	49%	-13%	14%	**5%**
3	52%	8%	**-8%**	23%	16%
4	46%	4%	**-5%**	25%	23%
5	17%	-17%	**-8%**	20%	20%
6	169%	92%	**-10%**	**10%**	**10%**

The accuracy of the used methods is summarized in TABLE XIV. , where the mean and standard deviation of the *absolute* relative errors are given for Real-Time projects, for non Real-Time projects and for the entire set of projects.

TABLE XIV. MEAN AND STDEV OF ABSOLUTE RELATIVE ERRORS

	NESMA ind. non norm.	NESMA ind. norm.	NESMA estim.	EQFP unspec.	EQFP generic
Mean (RT only)	112%	53%	10%	9%	8%
Stdev (RT only)	62%	42%	8%	8%	8%
Mean (non RT)	90%	41%	9%	17%	14%
Stdev (non RT)	61%	37%	3%	7%	7%
Mean (all)	102%	47%	10%	13%	11%
Stdev (all)	60%	38%	6%	9%	8%

VI. RELATED WORK

Meli and Santillo were among the first to recognize the need of comparing the various functional size methods proposed in the literature [18]. To this end, they also provided a benchmarking model.

In [12], van Heeringen et al. report the results of measuring 42 projects with the full-fledged, indicative and estimated NESMA methods. They found a 1.5% mean error of NESMA estimated method and a 16.5% mean error of NESMA indicative method.

Using a database of about 100 applications, NESMA did some research on the accuracy of the estimated and indicative function point counts. They got very good results (http://www.nesma.nl/section/fpa/earlyfpa.htm), although no statistics (e.g., mean relative error) are given.

In [16] Frank Vogelezang summarized the two techniques to simplified measuring given in the COSMIC measurement manual: the approximate technique (comparable to NESMA's indicative technique) and the refined approximate technique (comparable to NESMA's rough technique). In the approximate technique the average size of a functional process is multiplied with the number of functional processes the software should provide. In the refined approximate technique the functional processes to be provided can already be classified as small, medium, large or very large, each with its own average size. The precision of the COSMIC-FFP approximate technique is good enough with less than 10% deviation on a portfolio and less than 15% on a project within a specified environment [16].

Popović and Bojić compared different functional size measures –including NESMA indicative and estimated– by evaluating their accuracy in effort estimation in various phases of the development lifecycle [15]. Not surprisingly, they found that the NESMA indicative method provided the best accuracy at the beginning of the project. With respect to Popović and Bojić, we made two quite different choices: the accuracy of the method is evaluated against the actual size of the software product, not the required development effort, and –consistently– all the information needed to perform measurement is available to all processes.

There is no indication that real-time projects were among those measured by van Heeringen et al. or by NESMA.

VII. Conclusion

Sometimes, FPA is too slow or too expensive for practical usage. Moreover, FPA requires a knowledge of requirements that may not be available when the measures of size are required, i.e., at the very first stages of development, when development costs have to be estimated. To overcome these problems, simplified measurement processes have been proposed.

In this paper we applied simplified functional size measurement processes to both traditional software applications and Real-Time applications. The obtained results are fairly good, as a few of the tested methods provided average errors not greater than 13% (with fairly small standard deviations).

EQFP methods proved more accurate in estimating the size of non Real-Time applications, while the NESMA estimated method proved fairly good in estimating both Real-Time and non Real-Time applications. However, the relatively small number of projects involved in the analysis does not allow generalizing these results.

Even considering the relatively small dataset, it is however probably not casual that the NESMA estimated method happened to underestimate *all* projects. Probably NESMA should consider reviewing the weights used in the estimated method, in the sense of increasing them.

It is noticeable that all the used methods underestimated one of the Real-Time projects by over 20%. This can be quite dangerous, as underestimating size usually leads to dramatically underestimating the development effort, which can very easily cause the failure of the project. Our observations seem to suggest that the projects that are most likely to be underestimated by simplified methods are those characterized by the need to store or communicate many data at a time. The frequent occurrence of this condition should be checked before adopting a simplified measurement process.

Finally, we must point out that the results presented here are based on datasets in which the largest project has size of 289 FP. Further work for verifying the precision of simplified measurement methods when dealing with larger project is needed.

Among the future work is also the experimentation of simplified measurement processes in conjunction with measurement-oriented UML modeling, as described in [11].

Acknowledgment

The research presented in this paper has been partially supported by the project “Metodi, tecniche e strumenti per l’analisi, l’implementazione e la valutazione di sistemi software” funded by the Università degli Studi dell’Insubria

References

[1] A.J. Albrecht, Measuring Application Development Productivity, *Joint SHARE/ GUIDE/IBM Application Development Symposium*, 1979.

[2] International Function Point Users Group. Function Point Counting Practices Manual - Release 4.3.1, 2010.

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

[4] A.J. Albrecht and J.E. Gaffney, “Software function, lines of code and development effort prediction: a software science validation” IEEE Transactions on Software Engineering 9(6), 1983.

[5] L. Santillo, M. Conte and R. Meli, “Early & Quick function point: sizing more with less”, *Software Metrics, 2005. 11th IEEE International Symposium*, Como, 19-22 Sept. 2005.

[6] ISO, Iec 24570: 2004, Software Engineering-NESMA Functional Size Measurement Method version 2.1 - Definitions and Counting Guidelines for the Application of Function Point Analysis. International Organization for Standardization, Geneva, 2004.

[7] Early & Quick Function Points for IFPUG methods v. 3.1 Reference Manual 1.1, April 2012.

[8] L. Lavazza and C. Garavaglia, “Using Function Point in the Estimation of Real-Time Software: an Experience”, Software Measurement European Forum – SMEF 2008, Milano, 28-30 May 2008.

[9] L. Lavazza, V. del Bianco, C. Garavaglia, “Model-based Functional Size Measurement”, ESEM 2008, 2nd International Symposium on Empirical Software Engineering and Measurement, Incorporating ISESE and Metrics, Kaiserslautern, Germany. October 9-10, 2008.

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

[11] V. del Bianco, L. Lavazza, and S. Morasca, “A Proposal for Simplified Model-Based Cost Estimation Models”, 13th International Conference on Product-Focused Software Development and Process Improvement – PROFES 2012, Madrid, June 13-15, 2012.

[12] H. van Heeringen, E. van Gorp, and T. Prins, “Functional size measurement - Accuracy versus costs - Is it really worth it?”, Software Measurement European Forum – SMEF, Rome, 28 - 29 May 2009

[13] C. Jones, “A new business model for function point metrics”, http://www.itmpi.org/assets/base/images/itmpi/privaterooms/capersjones /FunctPtBusModel2008.pdf, 2008

[14] “Methods for Software Sizing – How to Decide which Method to Use”, Total Metrics, www.totalmetrics.com/function-point-resources/downloads/R185_Why-use-Function-Points.pdf, August 2007.

[15] J. Popović and D. Bojić, “A Comparative Evaluation of Effort Estimation Methods in the Software Life Cycle”, Computer Science and Information Systems, Vol. 9, n. 1, January 2012)

[16] F.W. Vogelezang, “COSMIC Full Function Points, the Next Generation”, in Measure! Knowledge! Action! – The NESMA anniversary book, NESMA, 2004.

[17] F.W. Vogelezang, A.J.E. Dekkers, “One year experience with COSMIC-FFP”, Software Measurement European Forum – SMEF 2004, January 28-30, Rome, 2004.

[18] R. Meli and L. Santillo, “Function point estimation methods: a comparative overview”, FESMA ’99, Amsterdam October, 4-8, 1999.