

An Empirical Evaluation of 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— 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 it requires more knowledge than available when development effort estimates are due. To overcome these problems, simplified methods have been proposed to measure Function Points. We used simplified methods for sizing both “traditional” and Real-Time applications, with the aim of evaluating the accuracy of the sizing with respect to full-fledged Function Point Analysis. To this end, a set of projects, which had already been measured by means of Function Point Analysis, have been measured using a few simplified processes, including those proposed by NESMA, the Early&Quick Function Points, the ISBSG average weights, and others; the resulting size measures were then compared. We also derived simplified size models by analyzing the dataset used for experimentations. In general, all the methods that provide predefined weights for all the transaction and data types identified in Function Point Analysis provided similar results, characterized by acceptable accuracy. On the contrary, methods that rely on just one of the elements that contribute to size tend to be quite inaccurate. In general, different methods show different accuracy for Real-Time and non Real-Time applications. The results of the analysis 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 details of both data and operations. Deriving our own models from the project datasets proved possible, and yielded results that are similar to those obtained via the methods proposed in the literature.

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

I. INTRODUCTION

The empirical evaluation of simplified Function Points processes [1] is motivated by the popularity of Function Points. In fact, Function Point Analysis (FPA) [2][3][4][5] is widely used. Among the reasons for the success of FPA is 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 relatively slow pace: between 400 and

600 function points (FP) per day, according to Capers Jones [6], between 200 and 300 function points per day according to experts from Total Metrics [7]. Consequently, 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 often 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 the aforementioned problems, simplified FP measurement processes have been proposed. A quite comprehensive list of such methods is given in [8]. Among these are the NESMA (NEtherland Software Metrics Association) indicative and estimated methods, and the Early & Quick Function Points method. Other methods were also proposed, including the Tichenor ILF Model [9] and models featuring fixed weights for the computation of size measures. These models are briefly described in Section II. 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 application of several simplified functional size measurement processes to 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, since 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, it has been shown that 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 [4] to size a set of non Real-Time programs, and we apply the guidelines given in [11] (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. In fact, there is little doubt that the simplified Functional Size Measurement (FSM) methods actually allow for early and quick sizing; the real point is to evaluate to what extent the savings in time and costs are paid in terms of inaccurate size estimates. So, we concentrate on the assessment of the accuracy of size estimates, for both Real-

Time and embedded applications, as well as “traditional” business applications. Throughout the paper, by “accuracy” we mean the closeness of a size estimate to the real size measure, i.e., the size measured according to IFPUG rules by an experienced measurer.

In this paper, we enhance the work reported in [1] by using an extended dataset, and by testing the usage of additional simplified FSM techniques, not used in [1]. However, in the paper we do not just evaluate existing proposals for simplifying the functional size measurement process; instead, we produce our own simplified models for estimating the functional size of software applications. This is done using the same approaches already used to produce the existing simplified methods: in fact, we obtained models that are structurally similar to the existing ones, but featuring different parameters (e.g., weights for basic functional components).

All the methods –i.e., both those proposed in the literature and ours– are tested on a set of projects and the results are compared.

We also analyze the differences between Real-Time and non Real-Time applications, and derive a few considerations on what models are best suited to estimate the size of each class of applications.

The results of the measurements and analyses reported in the paper are expected to provide two types of benefits: on the one hand, they contribute to enhancing our understanding of functional size measurement processes and their suitability; on the other hand, we provide useful information and suggestions to the practitioners that have to decide whether to use simplified FSM methods, and which one to choose.

The paper is organized as follows: Section II briefly introduces the simplified FSM 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. In Section VI, the dataset described in Section III is analyzed, in order to get simplified FSM models that are similar to those presented in Section II, but which rely on the measures of the considered projects. Section VII accounts for related work. Section VIII discusses the threats to the validity of the study. Finally, Section IX draws some conclusions and outlines future work.

Throughout the paper, we assume that the reader is familiar with the concepts of FPA and the IFPUG rules. Readers that need explanations and details about FP counting can refer to official documentation and manuals [4][5].

Throughout the paper, we refer exclusively to unadjusted function points (UFP), even when we talk generically of “Function Points” or “FP”.

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. Table III provides a quick overview of the activities required by FP measurement and estimation methods. Of course, the IFPUG method requires all the activities listed in Table III, while simplified methods require a subset of such activities.

A. Early & Quick Function Points

The most well-known approach for simplifying the process of FP counting is probably the Early & Quick Function Points (EQFP) method [12]. EQFP descends from the consideration that estimates are sometimes needed before requirements analysis 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 [4] are not used in EQFP, the result is a less accurate 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 [4][5], 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 Fig. 1 [12]).

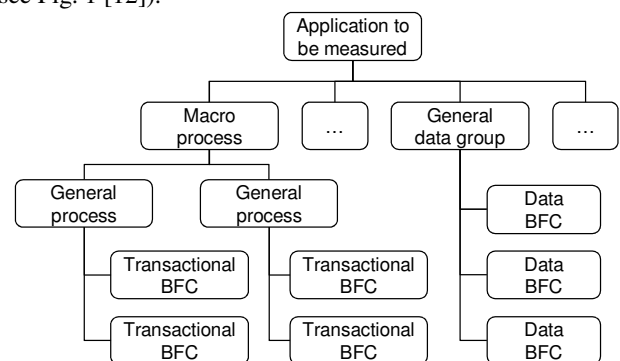


Figure 1. Functional hierarchy in the Early & Quick FP technique

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). 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 weight values for generic (i.e., not weighted) functions as given in [12]. The time and effort required by the weighting phases are thus 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	Weight		
	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 [12] are illustrated in Table II. When using this method, only the identification of logical data and elementary processes needs to be done: 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	Weight		
	Low	Likely	High
Unspecified Generic Processes	4.3	4.6	4.8
Unspecified Generic Data Group	6.4	7.0	7.8

B. NESMA indicative and estimated methods

The Indicative NESMA method [13] simplifies the process by only requiring the identification of LogicData from a conceptual data model. The Function Point size is then computed by applying the following formulae –where #ILF is the number of ILF and #EIF is the number of EIF– whose parameters depend on whether the data model is normalized in 3rd normal form:

Non normalized model: $FP = \# ILF \times 35 + \# EIF \times 15$

Normalized model: $FP = \# ILF \times 25 + \# EIF \times 10$

The process of applying the NESMA indicative method involves only identifying logic data and classifying them as ILF or EIF. Accordingly, it requires less time and effort than the EQFP methods described above, in general. However, 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 all assumed to be of low complexity, while Transactions Functions (EI, EQ and EO) are all assumed to be of average complexity:

$$UFP = \#EI \times 4 + \#EO \times 5 + \#EQ \times 4 + \#ILF \times 7 + \#EIF \times 5$$

C. Other simplified FSM process proposals

1) Tichenor method

The Tichenor ILF Model [9] bases the estimation of the size on the number of ILF via the following formula for transactional system (for batch systems, Tichenor proposes a smaller multiplier):

$$UFP = \#ILF \times 14.93$$

This model assumes a distribution of BFC with respect to ILF as follows: $EI/ILF = 0.33$, $EO/ILF = 0.39$, $EQ/ILF = 0.01$, $EIF/ILF = 0.1$. If the considered application features a different distribution, the estimation can be inaccurate.

The fact that a method based only on ILF requires a given distribution for the other BFC is not surprising. In fact, the size of the application depends on how many transactions are needed to elaborate those data, and the number of transaction cannot be guessed only on the basis of the number of ILF, as it depend on the number of ILF just very loosely. Instead of allowing the user to specify the number of transactions that are needed, the Tichenor method practically imposes that the number of transactions complies with the distribution given above.

2) ISBSG distribution model

The analysis of the ISBSG dataset yielded the following distribution of BFC contributions to the size in FP:

$$ILF 22.3\%, EIF 3.8\%, EI 37.2\%, EO 23.5\%, EQ 13.2\%$$

The analysis of the ISBSG dataset also shows that the average size of ILF is 7.4 UFP. It is thus possible to compute the estimated size on the basis of the number of ILF as follows:

$$UFP = (\#ILF \times 7.4) \times 100 / 22.3$$

The same considerations reported above for the Tichenor model apply. If the application to be measured does not fit the distribution assumed by the ISBSG distribution model, it is likely that the estimation will be inaccurate.

3) Simplified FP

The simplified FP (sFP) approach assumes that all BFC are of average complexity [14], thus:

$$UFP = \#EI \times 4 + \#EO \times 5 + \#EQ \times 4 + \#ILF \times 10 + \#EIF \times 7$$

4) ISBSG average weights

This model is based on the average weights for each BFC, as resulting from the analysis of the ISBSG dataset [15], which contains data from a few thousand projects. Accordingly, the ISBSG average weights model suggests that that the average function complexity is used for each BFC, thus

$$UFP = \#EI \times 4.3 + \#EO \times 5.4 + \#EQ \times 3.8 + \#ILF \times 7.4 + \#EIF \times 5.5.$$

TABLE III. ACTIVITIES REQUIRED BY DIFFERENT SIMPLIFIED MEASUREMENT PROCESSES

Measurement activities	IFPUG	NESMA indic.	NESMA estim.	EQFP Generic func.	EQFP Unspec. generic func.	Tichenor ILF Model	ISBSG distribution	sFP	ISBSG average weights
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	✓								

III. THE CASE STUDY

A. Real-Time projects

Most of the Real-Time projects measured are from a European organization that develops avionic applications, and other types of embedded and Real-Time applications. 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.

The projects' FUR were modeled using UML as described in [11], and then were measured according to IFPUG measurement rules [4]. 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 FPA, as described in [10]. The same projects were then sized using the simplified functional size measurement processes mentioned in Section II, using the data that were already available as a result of the IFPUG measurement.

Table IV reports the size in UFP 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.

B. Non Real-Time projects

The considered non Real-Time projects are mostly programs that allow users to play board or card games vs. remote players via the internet; a few ones are typical business information systems.

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 [16]; then, the function points were counted, on the basis of the model, according to IFPUG rules.

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

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

Project ID.	ILF	EIF	EI	EO	EQ	UFP
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
8	21 (3)	0 (0)	4 (1)	8 (2)	0 (0)	33
9	21 (3)	0 (0)	7 (2)	16 (4)	0 (0)	44

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

Project ID.	ILF	EIF	EI	EO	EQ	UFP
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
7	21 (3)	0 (0)	31 (9)	14 (3)	14 (4)	80
8	42 (6)	5 (1)	35 (9)	17 (3)	10 (2)	109
9	21 (3)	0 (0)	38 (11)	15 (5)	8 (2)	82

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.
- Similarly, the Tichenor ILF model and the ISBSG distribution models just require the ILF number.
- The NESMA estimated method, the EQFP generic functions method, the sFP method and the ISBSG average weights method 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 $21+2+5 = 28$.

TABLE VI. SIZES 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
8	33	105	75	35
9	44	105	75	49

A. Applying NESMA indicative

The applications to be measured were modeled according to the guidelines described in [16]. 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.

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.

TABLE VII. SIZES 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	92
2	94	200	140	93
3	79	120	85	88
4	96	140	100	111
5	90	105	75	102
6	91	245	175	93
7	80	105	75	88
8	109	225	160	106
9	82	105	75	98

C. Applying EQFP

As described in Figure 1., the EQFP method can be applied at different levels. Since we had the necessary data, we adopted the BFC aggregation level. 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 Table II are 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 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
8	33	35	38
9	44	49	52

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
7	80	95	92
8	109	113	113
9	82	104	103

TABLE X. MEASURES OF NON REAL-TIME PROJECTS OBTAINED VIA THE TICHENOR ILF MODEL, ISBSG DEISTRIBUTION, SFP AND ISBSG AVERAGE WEIGHTS METHODS.

Project ID	IFPUG	Tichenor ILF model	ISBSG distrib.	sFP	ISBSG average weights
1	92	90	199	112	98
2	94	60	133	113	100
3	79	45	100	99	91
4	96	60	133	123	118
5	90	45	100	111	109
6	91	105	232	114	98
7	80	45	100	97	92
8	109	90	199	126	112
9	82	45	100	107	104

TABLE XI. MEASURES OF REAL-TIME PROJECTS OBTAINED VIA THE TICHENOR ILF MODEL, ISBSG DEISTRIBUTION, SFP AND ISBSG AVERAGE WEIGHTS METHODS.

Project ID	IFPUG	Tichenor ILF model	ISBSG distrib.	sFP	ISBSG average weights
1	289	269	597	301	259
2	101	119	265	123	105
3	136	105	232	122	107
4	195	224	498	219	179
5	106	60	133	112	107
6	223	134	299	245	232
7	15	15	33	19	17
8	33	45	100	44	37
9	44	45	100	58	52

D. Applying Tichenor ILF Model

In order to apply the model we just had to multiply the number of ILF of each of our projects for the constant 14.93 suggested by Tichenor. The obtained results are illustrated in Table X and Table XI for non Real-Time and Real-Time projects, respectively.

When applying this method, it should be remembered that the results are likely to be incorrect if the distribution of BFC in the estimated application does not match the distribution observed by Tichenor. Accordingly, when applying the method, one should also check the distribution of BFC. Unfortunately, this implies making more work, namely, one should count the number of EIF, EI, EO, and EQ in addition to ILF. Even worse, one could discover that the distribution of his/her application is different from the distribution assumed by Tichenor, so that the estimated size is not reliable.

In our case, the projects do not appear to fit well in the distribution assumed by Tichenor: the *differences* between the measured ratios and the ratios expected by Tichenor are the following:

- For Real-Time projects: 14.3% for EI/ILF, 43.7% for EO/ILF, 3.9% for EQ/ILF, 7.9% for EIF/ILF.
- For non Real-Time projects: 96.7% for EI/ILF, 22.2% for EO/ILF, 27.3% for EQ/ILF, 14.7% for EIF/ILF.

In practice, our projects have a very different distribution of BFC sizes with respect to Tichenor expectations (for instance, in non Real-Time projects EI had often a larger size than ILF, while it is expected that the size of EI is about one third of the size of ILF). So, we must expect a quite poor accuracy from Tichenor estimates. This is actually confirmed by the data in Table XIV, Table XV and Table XVI.

E. Applying the ISBSG distribution model

We applied the formula $UFP = (\#ILF \times 7.4) \times 100 / 22.3$ prescribed by the method. Then, we evaluated the *differences* between the measured percentage contribution of BFC and the ISBSG averages. The differences we found were relatively small:

- For Real-Time projects: 28.7% for ILF, 3.4% for EIF, 19.3% for EI, 21.3% for EO, 13.2% for EQ.
- For non Real-Time projects: 12% for ILF, 4.8% for EIF, 5.6% for EI, 15.4% for EO, 13.2% for EQ.

Accordingly, we expect that the ISBSG distribution model applies reasonably well to our dataset, especially as non Real-Time projects are involved.

The obtained size estimates are illustrated in Table X and Table XI for non Real-Time and Real-Time projects, respectively.

F. Applying the sFP and ISBSG average weights

The application of the sFP and ISBSG average weights methods was extremely similar to the application of the NESMA estimated and EQFP generic methods, only the values of weights being different.

The obtained results are illustrated in Table X and Table XI for non Real-Time and Real-Time projects, respectively.

V. SUMMARY AND LESSONS LEARNED

In this section, the results of our empirical analysis are reports. First we discuss the quantitative results, then we analyze the results from a more theoretical point of view.

A. Applying the sFP and ISBSG average weights

To ease comparisons, all the size measures of RT projects are reported in Table XII and those of non RT projects are reported in Table XIII.

TABLE XII. 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	Tichenor ILF model	ISBSG distrib.	sFP	ISBSG average weights
1	289	645	460	245	262	262	269	597	301	259
2	101	280	200	99	102	106	119	265	123	105
3	136	245	175	101	100	108	105	232	122	107
4	195	570	405	168	181	182	224	498	219	179
5	106	140	100	100	102	106	60	133	112	107
6	223	330	235	216	208	229	134	299	245	232
7	15	35	25	16	16	17	15	33	19	17
8	33	105	75	35	35	38	45	100	44	37
9	44	105	75	49	49	52	45	100	58	52

TABLE XIII. 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	Tichenor ILF model	ISBSG distrib.	sFP	ISBSG average weights
1	92	225	160	92	100	99	90	199	112	98
2	94	200	140	93	107	99	60	133	113	100
3	79	120	85	88	97	92	45	100	99	91
4	96	140	100	111	120	118	60	133	123	118
5	90	105	75	102	108	108	45	100	111	109
6	91	245	175	93	100	100	105	232	114	98
7	80	105	75	88	95	92	45	100	97	92
8	109	225	160	106	113	113	90	199	126	112
9	82	105	75	98	104	103	45	100	107	104

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

Proj ID	NESMA ind. non norm.	NESMA ind. norm.	NESMA estim.	EQFP unspec.	EQFP generic	Tichenor ILF model	ISBSG distrib.	sFP	ISBSG average weights
1	123%	59%	-15%	-9%	-9%	-7%	107%	4%	-10%
2	177%	98%	-2%	1%	5%	18%	162%	22%	4%
3	80%	29%	-26%	-26%	-21%	-23%	71%	-10%	-21%
4	192%	108%	-14%	-7%	-7%	15%	155%	12%	-8%
5	32%	-6%	-6%	-4%	0%	-43%	25%	6%	1%
6	48%	5%	-3%	-7%	3%	-40%	34%	10%	4%
7	133%	67%	7%	7%	13%	0%	120%	27%	13%
8	218%	127%	6%	6%	15%	36%	203%	33%	12%
9	139%	70%	11%	11%	18%	2%	127%	32%	18%

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

Proj ID	NESMA ind. non norm.	NESMA ind. norm.	NESMA estim.	EQFP unspec.	EQFP generic	Tichenor ILF model	ISBSG distrib.	sFP	ISBSG average weights
1	145%	74%	0%	9%	8%	-2%	116%	22%	7%
2	113%	49%	-1%	14%	5%	-36%	41%	20%	6%
3	52%	8%	11%	23%	16%	-43%	27%	25%	15%
4	46%	4%	16%	25%	23%	-38%	39%	28%	23%
5	17%	-17%	13%	20%	20%	-50%	11%	23%	21%
6	169%	92%	2%	10%	10%	15%	155%	25%	8%
7	31%	-6%	10%	19%	15%	-44%	25%	21%	15%
8	106%	47%	-3%	4%	4%	-17%	83%	16%	3%
9	28%	-9%	20%	27%	26%	-45%	22%	30%	27%

TABLE XVI. MEAN AND STDEV OF ABSOLUTE RELATIVE ERRORS

	NESMA ind. non norm.	NESMA ind. norm.	NESMA estim.	EQFP unspec.	EQFP generic	Tichenor ILF model	ISBSG distrib.	sFP	ISBSG average weights
Mean (RT only)	127%	63%	10%	9%	10%	20%	112%	17%	10%
Stdev (RT only)	64%	44%	7%	7%	7%	16%	59%	11%	7%
Mean (non RT)	79%	34%	8%	17%	14%	32%	58%	23%	14%
Stdev (non RT)	56%	33%	7%	8%	8%	17%	50%	4%	8%
Mean (all)	103%	49%	9%	13%	12%	26%	85%	20%	12%
Stdev (all)	63%	40%	7%	8%	8%	17%	60%	9%	8%

The relative measurement errors are given in Table XIV and Table XV.

The obtained results show that we can divide the simplified FSM methods in two classes: those which base the size estimation exclusively on some measure of the data (like the NESMA indicative, the Tichenor and ISBSG distribution methods) and those which propose fixed weights for all the BFC of FPA.

The former methods yield the largest errors. Although it was expected that estimates based on less information are generally less accurate than estimates based on more information, the really important finding of our experimental evaluation is that the size estimates based on data measures feature quite often intolerably large errors, i.e., errors that are likely to cause troubles, if development plans were based on such estimates. For instance, let us consider the Tichenor method (which appears the best of those based on data measures) and assume that only size estimation errors not larger than 20% are acceptable: 10 estimates out of 18 would be unacceptable.

On the contrary, the methods that take into consideration all BFC and provide fixed weights for them yield size estimates that are close to the actual size. Among these methods sFP is an exception, since it regularly overestimates the size of projects, often by over 20%. This seems to indicate that

“average” projects are characterized by data and/or transactions whose actual complexity is smaller than the complexity expected by the sFP method.

The accuracy of the used methods is summarized in Table XVI, 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. The mean value of absolute relative errors is a quite popular statistic, often termed MMRE (Mean Magnitude of Relative Errors).

Table XVI shows that the NESMA estimated, the two EQFP methods and the ISBSG average weights methods provide essentially equivalent accuracy. This is not surprising, given that these methods propose very similar weight values. The NESMA estimated method appears the best, but for Real-Time projects the EQFP methods perform similarly, often even better.

For Real-Time projects, EQFP (either in the unspecified or generic flavor) tends to provide the most accurate results, while the NESMA estimated method provides quite reasonable estimates.

It is worthwhile noticing that EQFP is more accurate than NESMA for Real-Time applications because it uses bigger weights, which suite better Real-Time application, which are more complex than non Real-Time applications.

B. Theoretical analysis

As mentioned in Section II, simplified FSM methods are based on skipping one or more phases of the standards Function Point measurement process (see Table III). It is reasonable to assume that the accuracy of the measure is inversely proportional to the number of phases not performed, hence to the amount of data not retrieved from the functional user requirements of the software to be measured.

To confirm such hypothesis, we have enhanced the information reported in Table III with the data concerning mean errors and error standard deviations: the result is given in Table XVII. The direct comparison of accuracy data with the information used for measurement makes the following observations possible.

Any simplified method that does not involve the weighting appears to be bound to a 10-15% mean absolute error.

It does not appear true that the more you measure, the best accuracy you get. For instance, EQFP considering unspecified

generic functions appear more accurate than sFP, even though the former method does not involve classifying function types.

Among methods that use the same type and amount of data, there are relatively large differences in accuracy: for instance, the Tichenor ILF model appears more precise than both the NESMA indicative (with normalized data) and the ISBSG distribution.

The last two observations suggest that exploiting the knowledge provided by statistical analysis can be decisive for achieving accurate measures via simplified processes. For instance, the EQFP method considering unspecified generic functions is quite accurate because the likely complexity of data and transactions assumed by the method (see Table II) were derived via accurate statistical analysis. On the contrary, the complexity values assumed by the sFP method were chosen on the basis of expectations, not on rigorous statistical analysis.

The exploitation of statistical data is the base for the new methods described in the next section.

TABLE XVII. MEASUREMENT PROCESSES: REQUIRED DATA VS. ACCURACY

	IFPUG	NESMA indic. Norm.	NESMA estim.	EQFP Generic func.	EQFP Unspec. generic func.	Tichenor ILF Model	ISBSG distribution	sFP	ISBSG average weights
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	✓								
Mean error	-	49%	9%	13%	12%	26%	85%	20%	12%
Error stdev	-	40%	7%	8%	8%	17%	60%	9%	8%

(*) required to verify applicability

VI. NEW SIMPLIFIED FSM MODELS

In this section, we derive simplified FSM models similar to those described in Section II, but based on the measures of our own applications (as reported in Table IV and Table V).

In Table XVIII we give the average weights of the BFC computed over all the measured applications. Note that the given averages are computed as the mean –at the dataset level– of the mean values computed for each application. In the table, the mean weights derived from our dataset are shown together with the weights proposed by other simplified FSM methods, for comparison. The fact that our EI and EO means are smaller than the values proposed by other methods, while the ILF and EIF means are very close to those proposed by other methods probably means that our applications were simpler than those considered in the definition of other methods.

TABLE XVIII. AVERAGE FUNCTION TYPE WEIGHTS FOR OUR DATASET

Function type	EQFP generic	NESMA Estim.	ISBSG average	sFP	Our dataset (all proj.)
ILF	7.7	7	7.4	7	7.4
EIF	5.4	5	5.5	5	5.3
EI	4.2	4	4.3	3	3.7
EO	5.2	5	5.4	4	4.6
EQ	3.9	4	3.8	3	4

In Table XIX we give the average values of weights derived from our dataset, distinguishing Real-Time and non Real-Time applications. We also give the average value of the

ratio between the number of ILF and the size in UFP. It is possible to note that the average number of UFP per ILF we found is quite larger than that found by Tichenor. This suggests that models based just on ILF can be hardly generalized.

Note that we computed also the weights for transaction functions (TF) and data functions (DF). These weights can be used in simplified measurement processes like the EQFP unspecified generic method.

TABLE XIX. MEAN AND MDEIAN WEIGHTS FOR THE PROJECTS IN OUR DATASET

Dataset	Mean (median) weight							UFP/#ILF
	ILF	EIF	EI	EO	EQ	TF	DF	
All non RT proj	6.6	5.5	3.5	4.4	3.4	7.0	3.7	22.7
All RT proj	8.2	5.0	4.0	4.8	5.1	8.1	4.4	17.0
All proj	7.4	5.3	3.7	4.6	4.0	7.6	4.1	19.9

The values in Table XIX suggest that transactions were generally more complex in Real-Time applications than in non Real-Time applications. The latter are probably responsible for relatively smaller weights of transaction (EI, EO, and EQ) in Table XVIII.

Using the values in Table XIX it was possible to derive models that are similar to those described in Section II: they are described in Table XX and Table XXI.

TABLE XX. MODELS FOR NON RT PROJECTS.

Average weights (all BFC)	$UFP = 6.6 \#ILF + 5.5 \#EIF + 3.5 \#EI + 4.4 \#EO + 3.4 \#EQ$
Average weights (DF and TF)	$UFP = 7.0 \#TF + 3.7 \#DF$
ILF based model	$UFP = 22.7 \#ILF$

TABLE XXI. MODELS FOR RT PROJECTS.

Average weights (all BFC)	$UFP = 8.2 \#ILF + 5 \#EIF + 4 \#EI + 4.8 \#EO + 5.1 \#EQ$
Average weights (DF and TF)	$UFP = 8.1 \#TF + 4.4 \#DF$
ILF based model	$UFP = 17 \#ILF$

We used such models to estimate the size of the projects in our dataset. The results of the estimations are reported in Table XXII and Table XXIII for Real-Time and non Real-Time projects, respectively.

Table XXII and Table XXIII show a rather poor accuracy of the estimation based on ILF, with error greater than 20% for several projects.

On the contrary, the estimations based on average weights are reasonably accurate; the obtained results are particularly good for non Real-Time projects, with all the estimates featuring errors not greater than 10%.

The average values of the absolute relative errors are reported in Table XXIV together with the average values of the absolute relative errors obtained with the best among the other methods, for comparison.

It is easy to see that the estimates obtained using the average weights of the projects being estimated feature practically the same accuracy as the other methods.

TABLE XXII. ESTIMATES OF RT PROJECTS BASED ON MODELS USING THE PARAMETERS GIVEN IN TABLE XIX.

Proj. ID	Actual size	Average weights (all BFC)		Average weights (DF and TF)		ILF based model	
		Est. size	% err	Est. size	% err	Est. size	% err
1	289	273	-6%	277	-4%	306	6%
2	101	110	9%	109	8%	136	35%
3	136	109	-20%	105	-23%	119	-13%
4	195	187	-4%	198	2%	255	31%
5	106	104	-2%	103	-3%	68	-36%
6	223	223	0%	213	-4%	153	-31%
7	15	17	13%	17	13%	17	13%
8	33	39	18%	37	12%	51	55%
9	44	52	18%	51	16%	51	16%

TABLE XXIII. ESTIMATES OF NON RT PROJECTS BASED ON MODELS USING THE PARAMETERS GIVEN IN TABLE XIX.

Proj. ID	Actual size	Average weights (all BFC)		Average weights (DF and TF)		ILF based model	
		Est. size	% err	Est. size	% err	Est. size	% err
1	92	85	-8%	90	-2%	136	48%
2	94	87	-7%	97	3%	91	-3%
3	79	81	3%	84	6%	68	-14%
4	96	98	2%	102	6%	91	-5%
5	90	91	1%	92	2%	68	-24%
6	91	85	-7%	90	-1%	159	75%
7	80	79	-1%	79	-1%	68	-15%
8	109	98	-10%	101	-7%	136	25%
9	82	88	7%	88	7%	68	-17%

It is a bit surprising that in the literature a few models of type $UFP = k \times \#ILF$ were proposed, while model of type $UFP = k \times \#EP$ (where $\#EP$ is the number of elementary processes, i.e., $\#EI + \#EO + \#EQ$) received hardly any attention. We computed the ratio $UFP/\#EP$ for each application, and used the average value k in models $UFP = k \times \#EP$, to estimate the size of the applications in our dataset. The obtained estimates were characterized by errors quite similar to those of ILF-based models (the average absolute error was 25% for Real-Time projects and 27% for non Real-Time projects). Accordingly, it seems that models of type $UFP = k \times \#EP$ are not likely to provide good estimates.

TABLE XXIV. MEAN AND STDEV OF ABSOLUTE RELATIVE ERRORS

	Average weights, all BFC	Average weights, DF & TF	Average UFP / #ILF	NESMA estim.	EQFP unspec.	EQFP generic	ISBSG average weights
Mean (RT only)	10%	9%	26%	10%	9%	10%	10%
Stdev (RT only)	8%	10%	29%	7%	7%	7%	7%
Mean (non RT)	5%	4%	25%	8%	17%	14%	14%
Stdev (non RT)	3%	4%	22%	7%	8%	8%	8%
Mean (all)	8%	10%	31%	9%	13%	12%	12%
Stdev (all)	6%	6%	19%	7%	8%	8%	8%

VII. RELATED WORK

Meli and Santillo were among the first to recognize the need for comparing the various functional size methods proposed in the literature [17]. To this end, they also provided a benchmarking model.

In [18], 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 [19], Vogezeang summarized the two techniques to simplified measuring given in the COSMIC measurement manual: the approximate technique and the refined approximate technique. In the approximate technique, the average size of a functional process is multiplied with the number of functional processes the software should provide. The refined approximate technique uses the average sizes of small, medium, large and very large functional processes. The accuracy 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 [19].

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 [20]. 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 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.

In [8], Santillo suggested probabilistic approaches, where the measurer can indicate the minimum, medium and maximum weight of each BFC, together with the expected probability that the weight is actually minimum, medium or maximum. This leads to estimate not only the size, but also the probability that the actual size is equal to the estimate.

VIII. THREATS TO VALIDITY

A first possible threat to the internal validity of the study is due to the relatively small datasets.

Another possible issue concerns the size and complexity of the applications. As far as the Real-Time applications are concerned, we measured real industrial projects. Accordingly, we are fairly sure that they represent a good benchmark for the considered simplified FSM methods. On the contrary, our non Real-Time projects are fairly small. However, the really important point for testing the adequacy of simplified FSM methods is not the size of the benchmark applications, but their complexity. It is possible that our non Real-Time projects are slightly less complex than average applications: this would explain why most simplified FSM methods overestimate them (see Table XV).

The fact that our datasets are not very homogeneous is actually not a problem; rather it is useful to challenge the proposed simplified FSM methods with different types of software applications.

IX. 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 make it possible to draw a few relevant conclusions:

1. Some of the simplified FSM methods we experimented with seem to provide fairly good accuracy. In particular, NESMA estimated, EQFP, and ISBSG average weights yielded average absolute relative errors close to 10%. This level of error is a very good trade off, if you consider that it can be achieved without going through the expensive phase of weighting data and transactions.
2. Organizations that have historical data concerning previous projects can build their own models. We showed that with a relatively small number of projects it is possible to build models that provide a level of accuracy

very close to that of methods like NESMA estimated and EQFP.

3. The simplified FSM methods are generally based on average values of ratios among the elements of FP measurement. Accordingly, projects that have unusual characteristics tend to be ill suited for simplified size estimation. For instance, project 3 in our set of Real-Time projects is more complex than the other projects in the set, having most EI and EO characterized by high complexity. This causes most method to underestimate the size of the project by over 20%. Therefore, before applying a simplified FSM method to a given application, it is a good idea to verify that this application is not too much (or too less) complex with respect to “average” applications. Our Real-Time project 3 was characterized by the need to store or communicate many data at a time: this situation could have suggested that using average values for an early measurement leads to a rather large underestimation.

EQFP methods proved more accurate in estimating the size of 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.

When considering the results of our analysis from a practical viewpoint, a very interesting question is “what simplified method is the best one for my application(s)?”. Table XIV and Table XV show that the methods that are better on average are not necessarily the best ones for a given project. To answer the question above it would be useful to characterize the projects according to properties not considered in FSM, and look for correlations with the measures provided by different simplified methods. This would allow selecting the simplified measurement method that provided the best accuracy for applications of the same type as the one to be sized. Unfortunately, it was not possible to analyze the possibly relevant features of the dataset described in Section III (we had no access to the code of Real-Time projects), thus this analysis is among future activities.

As already mentioned, the results presented here are based on datasets in which the largest project has size of 289 FP: further work for verifying the accuracy 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 [16], as described in [21].

The models described in Section II are generally derived in a rather naive way, i.e., simply computing averages of some elements that are involved in the measurement: e.g., the average ration between the measure of BFC and their number. Simplified measurement models should be better derived via regression analysis. Unfortunately, the relatively little number of applications in our datasets does not support this type of

analysis, especially if multiple independent variables are involved, as in models of type $UFP = f(EI, EO, EQ, ILF, EIF)$ or $UFP = f(TF, DF)$. Performing this type of analysis is among our goal for future activities, provided that we can get enough data points.

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] L. Lavazza and G. Liu, “A Report on Using Simplified Function Point Measurement Processes”, Int. Conf. on Software Engineering Advances, (ICSEA 2012), Nov. 2012, pp. 18-25.
- [2] A.J. Albrecht, “Measuring Application Development Productivity”, Joint SHARE/ GUIDE/IBM Application Development Symposium, 1979.
- [3] 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, vol. 9, 1983.
- [4] International Function Point Users Group, “Function Point Counting Practices Manual - Release 4.3.1”, 2010.
- [5] ISO/IEC 20926: 2003, “Software engineering – IFPUG 4.1 Unadjusted functional size measurement method – Counting Practices Manual”, ISO, Geneva, 2003.
- [6] C. Jones, “A new business model for function point metrics”, <http://www.itmpi.org/assets/base/images/itmpi/privaterooms/capersjones/FunctPtBusModel2008.pdf>, 2008
- [7] “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.
- [8] L. Santillo, “Easy Function Points – ‘Smart’ Approximation Technique for the IFPUG and COSMIC Methods”, Joint Conf. of the 22nd Int. Workshop on Software Measurement and the 7th Int. Conf. on Software Process and Product Measurement, Oct. 2012.
- [9] C. Tichenor, “The IRS Development and Application of the Internal Logical File Model to Estimate Function Point Counts”, IFPUG Fall Conference of Use (ESCOM-ENCRESS 1998), May 1998.
- [10] L. Lavazza and C. Garavaglia, “Using Function Points to Measure and Estimate Real-Time and Embedded Software: Experiences and Guidelines”, 3rd Int. Symp. on Empirical SW Engineering and Measurement (ESEM 2009), Oct. 2009.
- [11] L. Lavazza and C. Garavaglia, “Using Function Point in the Estimation of Real-Time Software: an Experience”, Software Measurement European Forum (SMEF 2008), May 2008.
- [12] “Early & Quick Function Points for IFPUG methods v. 3.1 Reference Manual 1.1”, April 2012.
- [13] 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.
- [14] L. Bernstein and C. M. Yugas, “Trustworthy Systems Through Quantitative Software Engineering”, John Wiley & Sons, 2005.
- [15] International Software Benchmarking Standards Group, “Worldwide Software Development: The Benchmark, release 11”, 2009.
- [16] L. Lavazza, V. del Bianco, C. Garavaglia, “Model-based Functional Size Measurement”, 2nd International Symposium on Empirical Software Engineering and Measurement (ESEM 2008), Oct. 2008.
- [17] R. Meli and L. Santillo, “Function point estimation methods: a comparative overview”, Software Measurement European Forum (FESMA 1999), Oct. 1999.
- [18] 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 2009), May 2009.

- [19] F.W. Vogelezang, "COSMIC Full Function Points, the Next Generation", in *Measure! Knowledge! Action! – The NESMA anniversary book*, NESMA, 2004.
- [20] J. Popović and D. Bojić, "A Comparative Evaluation of Effort Estimation Methods in the Software Life Cycle", *Computer Science and Information Systems*, vol. 9, Jan. 2012.
- [21] V. del Bianco, L. Lavazza, and S. Morasca, "A Proposal for Simplified Model-Based Cost Estimation Models", 13th Int. Conf. on Product-Focused Software Development and Process Improvement (PROFES 2012), June 2012.