

An Exploratory Study on Functional Size Measurement based on Code

Hennie Huijgens
Delft University of Technology and
Goverdson, Delft, The Netherlands
h.k.m.huijgens@tudelft.nl

Magiel Bruntink
Software Improvement Group
Amsterdam, The Netherlands
m.bruntink@sig.eu

Arie van Deursen
Delft University of Technology
Delft, The Netherlands
Arie.vandeursen@tudelft.nl

Tijs van der Storm
Centrum Wiskunde & Informatica (CWI)
Amsterdam, The Netherlands
storm@cwi.nl

Frank Vogelezang
Ordina and COSMIC
Nieuwegein, The Netherlands
frank.vogelezang@ordina.nl

ABSTRACT

In this paper we explore opportunities, challenges, and obstacles that Functional Size Measurement (FSM) experts assume to be in automatically derived functional size, directly from the software project code itself. We designed a structured survey, that was answered by 336 FSM specialists. A majority of the respondents consider FSM to be an important tool for decision making. No indications are found for any perceived impact of agile methodology on the difficulty of applying FSM. Respondents overall think of automated FSM as important, but also difficult to realize. 54% of the respondents think that automated FSM will help measurement specialists, while 44% thinks that it will help decision makers too. The most preferred FSM method for automation is COSMIC (25%), followed by IFPUG (21%) and Nesma (16%). Respondents perceive automated FSM to be most suitable for baselining, benchmarking, and maintenance and legacy purposes.

CCS Concepts

• **General and reference** → **Cross-computing tools and techniques** → **Measurement**.

Keywords

Functional Size Measurement, FSM, automated FSM, Function Point Analysis, FPA, IFPUG, Nesma, COSMIC.

1. INTRODUCTION

Functional Size Measurement (FSM) has been widely accepted for decades as an early predictor of cost, duration and quality of software activities. FSM creates a context for software measurement based on the software's business value [1]. Among other attributes of software, size is one of the most significant [2].

At the same time FSM is accompanied by many limitations due to the manual counting effort needed, the often poor availability of reliable and correct functional documentation, and the sometimes

confusing translation of objective counting standards to the unruly practice in industry [3] [4].

The second of the four values mentioned in the Agile Manifesto [5] is “working software over comprehensive documentation”. Although its authors added the disclaimer “that is, while there is value in the items on the right, we value the items on the left more”, an often seen effect in agile practice is that the availability of a comprehensive set of reliable and correct functional design artefacts is simply missing. Meyer [6] labels the “depreciation of upfront tasks”, including functional design activities as “the undisputable prize winner of the bad and the ugly of agile approaches”. Yet how to perform FSM without reliable and available functional design artefacts?

The absence of reliable artefacts is not exclusively related to the agile domain. In fact, the shift towards agile approaches reveals a major shortcoming of FSM. The major source for FSM is a set of functional design artefacts. Thus low quality or bad availability of those artefacts will cause low quality FSM.

From this, we conclude that software measurement experts have a difficult time once companies go agile. This is particularly problematic when functional size measurement is used to normalize software activities, i.e., when size is used for estimation purposes or for benchmarking the performance of finalized software deliveries.

Summarizing we hypothesize that measurement experts face a dilemma where on one hand software companies recognize the need for FSM as a reliable tool for decision making on software investments. Yet on the other hand the counting process is experienced by software engineering practitioners as unreliable and time consuming due to a subjective element in interpretation of counting guidelines and the manual processing of sets of low quality functional design artefacts.

With this thought in mind, we raise the research question “*What do Functional Size Measurement experts assume to be the opportunities, challenges, and obstacles in deriving functional size directly, and in an automated way, from the software project code itself?*”

This paper is organized in the following way. Section 2 chalks out the backgrounds and related work on FSM. In Section 3 we describe our research method. The results are outlined in Section 4. Finally, Sections 5 and 6 include discussion, limitations, and threats to validity and conclusions and future work.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from Permissions@acm.org.
ICSSP'16, May 14-15, 2016, Austin, TX, USA
© 2016 ACM. ISBN 978-1-4503-4188-2/16/05...\$15.00
DOI: <http://dx.doi.org/10.1145/2904354.2904360>

2. BACKGROUND AND RELATED WORK

FSM origins from function point analysis (FPA), designed by Albrecht in 1979 [7] to estimate size of software delivery by means of user functionality. FSM is based on the complete set of functional requirements of a software project or a software system. An extensive overview of FSM can be found in [2] [1] and [4].

FSM is an industry standard to measure size of software engineering activities. With ISO/IEC 14143 as an umbrella standard, five FSM methods are certified by ISO as an international standard:

1. ISO/IEC 19761:2011: COSMIC FSM method [8];
2. ISO/IEC 20926:2009: IFPUG FSM method [9];
3. ISO/IEC 20968:2002: MkII FPA FSM method [10];
4. ISO/IEC 24570:2005: Nesma FSM method version 2.1 [11];
5. ISO/IEC 29881:2010: FiSMA FSM method version 1.1 [12].

Three of the above mentioned ISO standards are commonly used: IFPUG, COSMIC, and especially in The Netherlands Nesma. In the remaining of this study we focus at these. Although IFPUG and Nesma counting rules are often assumed to be equivalent, we decided to use both standards in our study. The main reason for this is that we focus at the Nesma (detailed) method, but also at the additional counting guidelines for the so-called estimated approach (a high level approach where all logical files are counted with complexity level low, and all user transactions are counted with complexity level average) [11]. The counting rules for these standards are maintained by three FSM associations: The International Function Point User Group (IFPUG), the Netherlands Software Measurement Users association (Nesma), and the Common Software Measurement International Consortium (COSMIC).

Automated FSM based on the IFPUG method was inventoried in a 1996 software tool market survey [13]. It mentions eight tools that measure FPs directly from functional requirements models (e.g. data flow diagrams, entity-relationship diagrams, or object models), but their accuracy has not been independently validated and they provide no insight in applied measurement algorithms. Other efforts on automated FSM on IFPUG counting rules are a framework based on a to be build slicing tool for automated counting of IFPUG function points in COBOL source code [14]. More recently, the Object Management Group (OMG) developed a standard on Automated Function Points (AFP) based on the IFPUG method [15], that is relatively widely supported in industry, among others by CAST Software tools [16] [17]. The OMG approach is analysed and discussed in [18].

Automated generation of functional size based on design artefacts, such as UML models [19] [20] [21] [22], OO models [23] [24], or user interface formats [25], of which a majority is focused at the COSMIC method [26] [27] [28]. An overview of procedures that use conceptual models as basis for functional size is given in [29]. Practical implementations of automated COSMIC FSM based on functional design artefacts are described in [30] and [31].

With regard to what FSM method to use, Živković et al. [4] argues that MK II has some advantages compared to IFPUG, notably when a lot of DETs are present. However, both methods performed poorly in the case of real-time applications and system software; COSMIC gives better results with a higher number of FPs [4].

3. RESEARCH METHOD

In this section we describe our research questions and the method that we applied for our study.

3.1 Research Questions

Our examination of existing literature revealed that no open source solution for automated FSM is available. A limited number of - poorly documented - implementations of AFP is available in industry, including a commercial solution (CAST) that delivers FPs based on IFPUG counting rules. The applicable algorithm is partly documented in the OMG standards on automated FSM [15]. At the same time agile is changing the world of software developers. Its rapid stream of iterations and changing focus from estimation towards analysis asks for measurement support tools that are code-based instead of design-based and deliver fast and reliable functional size measures.

This study is a first, exploratory step in possible future research on automated FSM. Our goal is to help define a long-term vision on a solution, or a series of solutions, to automatically derive FSM from source code written in a number of widely used computer languages. A to be expected side-effect of our research is the creation of a comprehensive set of functional design artefacts of a software system or a software project that is automatically derived from its source code, however future research should determine the scope of this.

Consequently, we developed the following research questions in order to gain an in-depth understanding of the working practices and challenges of FSM specialists and opportunities with regard to automation of FSM based on code:

- RQ1 Is FSM (still) considered an important decision making tool?*
- RQ2 Is there any perceived impact of agile methodology on the difficulty of applying FSM?*
- RQ3 To what extent is the automation of FSM considered an important step, and to what extent is it perceived difficult or impossible?*
- RQ4 To what extent are current FSM (automation) tools and related approaches (e.g. backfiring) serving the needs to FSM specialists?*

3.2 FSM Expert Survey Design

In order to find an answer to our research questions we performed an exploratory study among communities of FSM experts. Our study methodology involves a quantitative survey that includes qualitative open questions.

Protocol: We created a 25-minute survey focused at software measurement experts in industry and in government organizations. We ask the participants to rate their agreement with a number of propositions on opportunities, challenges, and obstacles with regard to automated deduction of functional size from a projects or a systems source code, without the use of functional design documentation. A separate Technical Report [32] gives a comprehensive inventory of survey questions and options, yet the following overview summarizes the survey:

1. The survey collects demographic information and some basic understanding of the professional background of participants (type of organization, main role, experience level in FSM, membership of FSM-communities, certification for FSM-methods).
2. In what measure do you agree with the following statements on the overall importance of FSM? The two statements were randomized. Besides ratings on a 1-5 Likert scale we ask the participants to add free format text as an explanation of their perceptions:

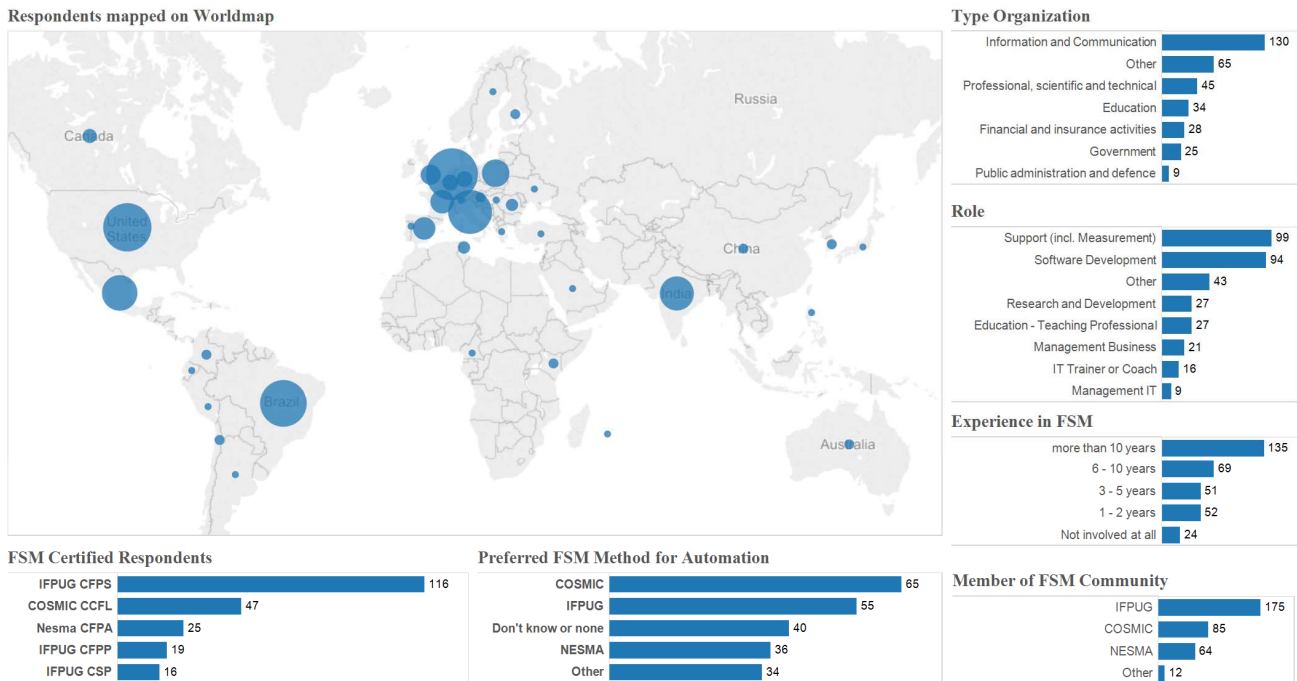


Figure 1. Overview of survey respondent’s demographics. The size of the circles indicates the number of respondents.

- a. Functional Size Measurement is an important tool for decision makers on software projects.
- b. Agile software development hinders the preparation of good and reliable FSM.
3. What factors were involved in your own organization that contributed to success or failure of projects that used FSM in an agile context? (Free format text question).
4. Which of the following approaches related to FSM do you use in practice? (Select all that apply).
5. Which of the following FSM methods would you rate as most suitable for automation based on source code (IFPUG, Nesma, COSMIC, commercial or self-developed tools, backfiring)?
6. To what extent do you agree with the following statements? See Table 1 for an overview of the seven statements. The statements are randomized. Besides ratings on a 1-5 Likert scale we ask the participants to add free format text as an explanation of their perceptions.
7. For what purposes do you think Automated FSM based on Code is most suitable? For what purposes do you think Automated FSM based on Code is not suitable? (Free format text).

Participants: We recruited, in close cooperation with the applicable boards, a range of measurement specialists that are connected to three major associations in the field of FSM: IFPUG, Nesma, and COSMIC.

3.3 FSM Expert Study Analysis

We examine 1-5 Likert distributions for the respondent set as a whole and for a number of subsets by using a Wilcoxon rank-sum test. We compute the mean and the standard deviation for each

question that is based on a 1-5 Likert scale. Subsequently we calculate indicators that might help us to interpret the results of the survey (see Table 1):

- *Top-Box*: the percentage respondents that strongly agreed.
- *Top-2-Box* or the percent agree; the percentage respondents that agreed or strongly agreed.
- *Net-Top-2-Box*; the percentage respondents that chose the top 2 bottom responses subtracted from the top-2 top responses.
- *Coefficient of Variation (CV)*; also known as relative standard deviation; the standard deviation divided by the mean. Higher values indicate higher variability.

Where the first three are measures of the central tendency, CV is a measure of variability; we use it in addition to the other approaches. In order to examine whether the free format text resulting from the survey confirms observations from the quantitative analysis we code the free text from the survey using Qualyzer¹.

3.4 Demographics of Survey Respondents

An invitation letter for our survey was sent by the boards of IFPUG, Nesma and COSMIC to people in their mailing lists. Besides that, we asked FSM specialists to answer the survey via social media. The survey has been completed by 336 respondents from 40 different countries (see Figure 1). Not all survey questions were answered by all respondents. However, for 336 respondents enough answers were applicable to include them in the analysis. The countries from which most completed surveys have been received are The Netherlands (53 respondents), United States (46), Brazil (43), Italy (38), Mexico (25), and India (23).

¹ <http://qualyzer.bitbucket.org>.

Table 1. An overview of the overall results of the rating questions in the survey.










Question	Likert Distribution	Number of respondents	Mean	Percent Agree	Top-Box	Net-Top-2-Box	Coefficient of Variance
Functional Size Measurement is an important tool for decision makers on software projects		245	4.27	87%	47%	81%	20%
The tool(s) that I use for Functional Size Measurement satisfies my company's needs		56	3.64	59%	21%	45%	28%
Automated derivation of <name of FSM-method> directly from source code is difficult		211	3.54	50%	21%	36%	30%
Automated derivation of <name of FSM-method> directly from source code will help measurement specialists		211	3.36	54%	12%	33%	33%
Automated derivation of <name of FSM-method> directly from source code is important		211	3.20	42%	9%	21%	33%
Automated derivation of <name of FSM-method> directly from source code will help decision makers on software projects		211	3.19	44%	9%	20%	34%
Agile software development approaches hinder preparation of good and reliable functional size measurements		245	2.69	22%	4%	-20%	40%
Backfiring is a reliable measurement tool for conversion of Lines of Code data into Functional Size data		39	2.69	23%	3%	-23%	37%
Backfiring is a reliable measurement tool for conversion of Functional Size data into Lines of Code data		39	2.54	23%	3%	-28%	44%

Table is sorted by Mean. When in a question the variable <name of FSM-method> is included, the applicable name of the FSM-method selected as 'most suitable for automation' was shown. Column 'Likert Distribution' shows a graph of the distribution on a 1-5 point Likert scale for each question with from left to right the values 'Strongly disagree', 'Disagree', 'Neutral', 'Agree' and 'Strongly agree'.

Respondents have different organizational backgrounds. Most (39%) work for Information and Communication companies, 13% perform professional, scientific and technical activities, and 10% come from education. Respondents fulfil different roles. Most (29%) work as an ICT Professional - Support (including Measurement and Analysis). 27% works as an ICT Professional - Software and Applications Development and Analysis. 8% works as a teaching professional, while a same percentage works as a researcher. 14% of the respondents have a role as a manager, where we assume these to be decision makers with regard to our further analysis.

With regard to their level of experience the survey indicates that respondents tend to be involved in FSM for a longer time. 41% is involved in FSM for more than 10 years, 21% for 6 to 10 years, while only 16% is a starter in the field of FSM. We included data of all respondents – whether experienced or not in FSM – in our analysis, but we specifically looked at differences between both groups.

244 Of the respondents (73%) are a member of one or more FSM communities, such as IFPUG (54%), COSMIC (26%), or Nesma (20%). A limited number of respondents are a member of other functional size-related communities, such as Gartner, CAST, GUFPI-ISMA, DASMA, SiFPA, ASSEMI, or ISBSG. Of the group of 336 respondents 183 (54%) are certified for one or more FSM-methods; either COSMIC-CCFL (47), IFPUG-CFPP (19), IFPUG-CFPS (116), or Nesma-CFPA (25).

4. RESULTS

This section presents the results of our exploratory survey among FSM specialists. When quoting survey respondents, we refer to the individual contributor using a [RX] notation, where X is the answer's ID. We present codes resulting from coding open-ended answers as lists with the percentage of each code between brackets. Survey results are summarized in Table 1.

4.1 RQ1: Importance of FSM

Not to our surprise, a vast majority of respondents (87%) agrees with the statement that FSM is an important tool for decision makers. A high Net-Top-2-Box of 81% in combination with a low Coefficient of Variation of 20% indicates a shared opinion on this. Analysis of free format text confirms this observation. Coding resulted in the following most mentioned reasons, where the percentage behind each item indicates the proportion of a specific code versus all codes applied on remarks:

1. FSM supports effort, cost, and time estimation (26%).
2. FSM supports benchmarking (20%).
3. FSM supports decision making (19%).
4. FSM is objective (15%).
5. FSM enables reliable planning and budgeting (8%).

Many respondents emphasize their opinion that software size is the single most important factor in software cost estimation, and that FSM is the only and best method to count software size: “*FSM methods supply objective size of the project, not influenced by implementation technology or team experience*” [R244]. “*Functional Size is excellent base for Total Cost of Investment and Total Cost of Ownership estimation*” [R085]. “*Today Functional Points is the better and most structured measurement method to projects in general*” [R030]. “*When using other people's money, it's important to have some capability of telling them what it might cost*” [R012].

As Table 2 shows, respondents of different sub-selections (e.g. certified or non-certified respondents, respondents with a business, or an IT role, respondents that are a member of IFPUG, Nesma, or COSMIC) do overall agree on this statement; the relatively low variance indicates that the means of each sub-selection are closely within one range.

Table 2. Variance of Respondents Answers.

Research Question	Variance
RQ4 - Satisfied with FSM Tools?	0.1520
RQ4 - Is backfiring reliable? (FSM to LOC)	0.1277
RQ4 - Is backfiring reliable? (LOC to FSM)	0.0671
RQ3 - Is automated FSM difficult to realize?	0.0608
RQ3 - Does automated FSM help decision makers?	0.0584
RQ3 - Is automated FSM important?	0.0506
RQ3 - Does automated FSM help FSM specialists?	0.0421
RQ1 - Is FSM important?	0.0228
RQ2 - Does agile hinder FSM?	0.0152

Table is sorted by Variance. Variance is calculated for each research question on the means of the following sub-selections of respondent answers: certified, non-certified; role business, IT, and other; member IFPUG, Nesma, COSMIC; and preference IFPUG, Nesma, COSMIC, and other.

4.2 RQ2: Impact of agile on FSM

Our assumption that agile delivery models tend to hinder FSM is not confirmed by a majority of the respondents. Our assumption that, due to the assumed depreciation of upfront tasks, a lack of proper design documents blocks reliable FSM seems not true. Although more respondents do disagree than agree on this (Net-Top-2-Box is -20%), a high CV score (40%) indicates different (and many neutral) opinions: *“Agile is a development approach like any other”* [R066]. Analysis of the free format text resulted in the following most mentioned reasons by respondents that agree on the statement that agile hinders FSM:

1. Poor documentation in agile (10%).
2. Open scope and changing requirements (6%).
3. Short cycle of agile does not fit with FSM (3%).
4. FSM fits waterfall better (3%).
5. Developers do not like disturbance for FSM (3%).

The free format text supports that FSM fits with agile, yet only when performed after finalization of a sprint and not for estimation purposes: *“it all depends when you count a piece of functionality. If a piece of functionality is changed frequently, there might be something wrong with the requirements. If at end of a project or major release, then okay”* [R016]. Other respondents did agree with the fact that agile and FSM do not always fit together: *“No detailed documentation system requirements traceability becomes quite costly due to the speed of evolution of the system’s features to be measured”* [R020]. *“The experience we had with software development using agile approach to government was terrible. All artefacts produced were disapproved by the team of quality assurance and the customer”* [R037].

The following reasons against the statement that agile development hinders FSM were mentioned by opponents:

1. FSM is possible in agile when implemented properly (28%).
2. FSM is independent from a development method (14%).
3. Documentation is a maturity issue (7%).
4. Good experience with FSM and agile (6%).
5. FSM is possible without detailed documentation (3%).

“As long as requirements are clear and scope is defined you can get a functional size measurement; maybe not so perfect but close enough” [R005]. *“It depends on the maturity of the staff in documenting what is necessary to give a functional view. The staff also*

has to think functionally. It is better when you have business specialist in the staff, not only technical professionals” [R026].

In order to better understand the backgrounds of respondents answers with regard to the application of FSM in combination with agile development, we asked them “What factors were involved in your own organization that contributed to success or failure of projects that used Functional Size Measurement in an agile context?” Negative factors that were mentioned are ‘FSM is not applied in agile or with many problems’, ‘limited functional documentation’, and ‘limited knowledge and resistance in agile teams against FSM’. *“FSM is too much related to waterfall”* [R080]. *“Lack of awareness that productivity measurement is important to use for new bids”* [R082]. *“need to keep a close watch on the requirements which are part of multiple sprints and count the requirements only once”* [R098].

Positive factors are the actual use of FSM in agile, many examples of success factors yet no umbrella aspects; ‘estimate scope upfront’ and ‘monitor progress after sprint’; and ‘commitment of upper management’. *“Rigorous process for requirements management and measurement based retrospectives connected to unit pricing (cost per FP)”* [R075]. *“Mapping a user story to a functional artefact”* [R169]. *“Functional size is a good basis to establish project budgets. Even in an agile approach one has to decide on the budgets required to end up with a set of useful products”* [R191].

A relatively low variance in Table 2 for this research question indicates that overall respondents from sub-selections did agree on their opinions.

4.3 RQ3: Automation of FSM

In order to gain insight into the backgrounds of RQ3 ‘To what extent is the automation of FSM considered an important step, and to what extent is it perceived difficult or impossible?’, we asked the respondents to give their opinion on five aspects of automation.

4.3.1 Preferred FSM method for automation

We asked the respondents “Which of the following Functional Size Measurement methods would you rate as most suitable for automation based on source code?” The FSM method that apparently is preferred for automation by most respondents is COSMIC; 25% of the respondents opted for this method. IFPUG was chosen second best with 21%. Nesma was picked last with 16%, of which 14% was labelled at the so-called estimated approach (a high level approach where all logical files are counted with complexity level low, and all user transactions are counted with complexity level average). 34% of the options selected were of the label “Other”. In the free format text, the following clarifications were given: any option, backfiring, CAST, FFPA Gartner, IFPUG estimated approach, OMG, and Simple Function Points (22%). 9% of the respondents opted for “none”, indicating that they did not believe that automation is preferred or possible: *“I believe that automation from source code is so highly dependent on programming styles as to make it unsuitable for general use”* [R243]. 7% indicated not to be able to answer this question: *“Cannot judge at this point”* [R065], *“I don’t know any, but I start with COSMIC proximately”* [R139].

In the following survey questions, that were performed in a randomized order, the respondents were asked to rate questions in relation to the FSM method that they rated as most suitable in RQ3.

4.3.2 The importance of automated FSM

A majority of respondents is neutral on the questions whether automated FSM is important. 42% agrees on this statement, yet

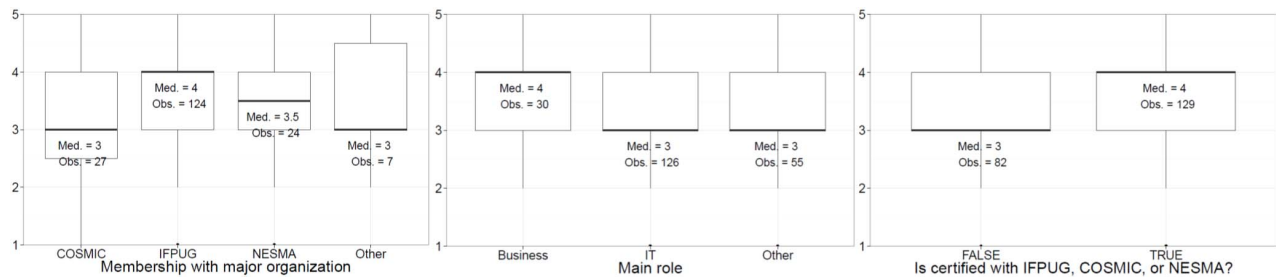


Figure 2. Boxplots on different sub-selections of respondents with regard to the question ‘Automated derivation of <preferred method> directly from source code is difficult’.

apparently many are uncertain (neutral) about this looking at the low Net-Top-2-Box and high Coefficient of Variation scores. Analysis of free format text revealed the following most mentioned reasons why automated FSM is important:

1. More reliable than backfiring (17%).
2. Will help decision making (13%).
3. Will be faster (12%).
4. More reliable and accurate (9%).
5. Saves manual effort (6%).

We assume that the rightly or wrongly expected fact that automation always leads to “faster, better, and more reliable” might play a role here. In a way the answers might indicate somewhat false expectations: “It will be important if it is accurate” [R065]. “The time and cost of function point counting is always an issue. Reducing this would be a big help” [R121]. The following most mentioned reasons illustrate why respondents are neutral on automated FSM or assume it not to be important:

1. Questions on the added value of automated FSM (10%).
2. Accuracy is an issue (9%).
3. Doubts on whether automated FSM is feasible (6%).
4. FSM should not be derived from technical size at all (5%).
5. Automated FSM is not useful for estimation (to late) (4%).

As mentioned before, not all respondents were convinced of the idea of automation of FSM based on code: “Who cares after the facts?” [R185]. “It could be false” [R128]. “It is important but not from source code” [R100]. A relatively low variance (see Table 2) indicates that respondents in sub-selections more or less agreed on the outcomes with regard to this research question.

4.3.3 Is automated FSM difficult to realize?

In line with the results on importance of automated FSM a vast majority of respondents (50%) expects it to be difficult, while many are neutral on the question whether this idea will be difficult. The following most mentioned reasons illustrate why respondents assume automated FSM to be difficult or why they are uncertain (neutral) about this statement:

1. Functional and technical are different views (26%).
2. Complexity and variation in source code (14%).
3. Large number of programming languages (10%).
4. Difference in technologies, architectures, and skills (10%).
5. Doubts on accuracy and reliability (6%).

Several difficulties were foreseen here, where many are related to difficulties related to translation of technical items to functional objects: “Automation of counting from the COSMIC would involve the identification of functional processes and objects of interest, that can be considered hard to identify by their nature “stochastic” [R133]. “Because Nesma is functionally oriented and not technically, as in source code” [R012]. Others relate to quality of source code: “Because of the poor quality of the code source. I think there may be a large deviation of the actual size and derived from source code” [R132]. And some just think it is impossible to do; “I think it can't be done” [R112].

While it might be clear from Table 1 that not many respondents disagreed with the above statement it is remarkable that still 13% states that automated FSM can be achieved without major difficulties. “The structure of COSMIC is similar to the structure of code” [R053]. “A tool has been proposed for the C language at ESTACA” [R022]. “There are tools for converting source code to UML; from that you can generate new code” [R235].

Table 2 shows a relatively high variance, indicating respondents in different sub-selections did not all agree on this question. Within members of different FSM communities (IFPUG, Nesma, and COSMIC) relatively large differences occur (see Figure 2). Apparently COSMIC members judge automation of FSM based on code to be less difficult than both Nesma and IFPUG members. Looking at differences between roles, we notice that respondents with a business role apparently think that automation is easier than respondents with an IT or other role. Finally, it shows that certified respondents think that automation is more difficult than non-certified respondents do.

A remark is in place with regard to the assumed difficulty of automation. Within the scope of this exploratory study we did not analyse any technical insights on the actual difficulties behind such an automation (except the experts’ opinions that this is difficult).

4.3.4 Does automated FSM help?

A bit more than half of the respondents (Top-2-Box 54%) think that automated FSM will help measurement specialists. Less than half of the respondents (Top-2-Box 44%) think that it helps decision makers. Many respondents are neutral on both aspects, a minority does not agree with these statements. A strong positive correlation is found between these statements and ‘Automated FSM is important’ (see Table 3). We assume that respondents that rate automated FSM as important do so because they think it helps both measurement specialists and decision makers. Analysis of the free format text shows that respondents mentioned the following reasons why automated FSM will help measurement experts:

Table 3. Matrix with test results of association between paired samples, using Kendall's tau Rank Correlation.

	Agile hinders	Automation important	Automation difficult	Automation will help specialists	Automation will help decision makers	Current tool satisfactory	Backfiring reliable (code to FSM)	Backfiring reliable (FSM to code)
FSM importance	-0.28 (0.00)	0.04 (0.58)	0.32 (0.00)	-0.12 (0.08)	0.05 (0.50)	0.20 (0.15)	-0.06 (0.73)	0.01 (0.97)
Agile hinders		-0.17 (0.01)	-0.04 (0.53)	-0.07 (0.33)	-0.11 (0.11)	-0.19 (0.16)	-0.17 (0.30)	-0.11 (0.53)
Automation important			-0.24 (0.00)	0.76 (0.00)	0.80 (0.00)	-0.01 (0.97)	0.36 (0.02)	0.40 (0.01)
Automation difficult				-0.31 (0.00)	-0.18 (0.01)	0.07 (0.62)	-0.30 (0.10)	-0.16 (0.34)
Automation will help specialists					0.80 (0.00)	-0.03 (0.80)	0.27 (0.10)	0.21 (0.20)
Automation will help decision makers						0.03 (0.80)	0.19 (0.25)	0.26 (0.09)
Current tool satisfactory							-0.28 (0.21)	-0.25 (0.26)
Backfiring reliable (code to FSM)								0.80 (0.00)

The table above shows results from a test of association between paired samples of the survey results, using Kendall's tau Rank Correlation. The overview shows for each test the correlation coefficient and between brackets the p-value. A green color indicates samples that show a strong positive and significant linear relationship.

1. Faster and cheaper measurements (23%).
2. Improves the quality of measurements (11%).
3. Supports baselining and benchmarking (10%).
4. Measurement experts focus on exceptions and learning (10%).
5. Compare and validate estimations and realization (8%).

Alike the earlier question on importance of automated FSM, we assume that referring to automation as such, leads to expectations that FSM will be faster, better, and cheaper by default: "Measurement will be very fast and easy to do" [R304]. Although, comments indicate that automation will be a big help for experts: "Measuring COSMIC at a detailed level is cumbersome work. If this part can be automated, that would be great news for measurers" [R053]. "Overcomes one of the biggest barriers to entry - allows companies with limited resources (i.e., no Certified Function Points Specialists, limited budget) to size their portfolio quickly and easily" [R260]. "If automated derivation means that more benchmarking will be done, it would give us much more information on realized projects" [R267]. With regard to respondents arguing against the above statement: 9% expect quality issues and 6% has doubts whether automated FSM will help measurement specialists in any way: "From my numerous years of experience and after reviewing tools that claim to automate the counting of Functional Size, I have found they are unreliable and not accurate" [R157]. The following reasons were mentioned with regard to the question whether automated FSM helps decision makers:

1. Faster and cheaper measurements (21%).

2. Enables better decision making (13%).
3. Supports baselining and benchmarking (13%).
4. Evolutionary maintenance in agile environments (11%).
5. Improves quality of measurements (4%).

Arguing against the statement that automated FSM helps decision makers, 13% of the respondents mention that it does not support upfront estimation due to the fact that no code is available at that stage: "It is most likely that FSM should be done before source code exists" [R096]. "okay for baseline assessment" [R121]. 12% expects quality issues: "I doubt if it's reliable enough for measurement specialists" [R112] "Won't be available early on when decisions need to be made" [R242]. 5% doubts whether automated FSM will help decision makers in any way.

4.3.5 Purposes of automated FSM

We asked the respondents 'For what purposes do you think Automated Functional Size Measurement based on Code is most suitable?' Analysis of free format text revealed the following aspects where automated FSM is expected to be successful.

1. Application and portfolio sizing (baseline) (23%).
2. Build historical database and benchmarking (16%).
3. Supports maintenance and legacy (15%).
4. Support (large-scale application) estimation (13%).
5. Supplier management and outsourcing (7%).

As the inventory shows, most mentioned were aspects related to baselining applications or portfolios as a whole, where we assume this to be closely related to the second aspects benchmarking and the third with regard the suitability for maintenance and legacy. "All purposes for which completed applications are available; application portfolio sizing, application management contracting, building historical performance data of your own organization" [R089]. "Application sizing for Maintenance assessments" [R209]. "Baseline estimation for big amounts of source code which have never been measured before" [R299]. "Legacy systems without documentation" [R047]. A number of respondents indicate that automated FSM can support estimations. Although, most of them emphasize the purpose for existing or very large systems: "Very fast estimations for very big projects" [R144]. "Want to estimate on the project size of an existing system" [R263]. Surprisingly enough some respondents see automated FSM as additional to manual counting: "As a second opinion in addition to a manual count" [R274].

Secondly, we asked 'For what purposes do you think Automated Functional Size Measurement based on Code is not suitable?' Free text analysis resulted in some findings, although not many aspects were mentioned.

1. Pre-build estimation (42%).
2. Detailed FSM calculations (16%).
3. Productivity analysis afterwards (16%).
4. Benchmarking (11%).
5. Accurate and consistent FSM (11%).

Where large scale application estimation is perceived to be a suitable purpose for automated FSM, the opposite is the case for upfront estimation, since no code is available at that moment: "For estimated measurements before code is available" [R197]. A number of respondents mentioned that automated FSM is not to be used in contract negotiations where detailed FSM is obliged. "Good estimations with enough detail to calculate the final effort

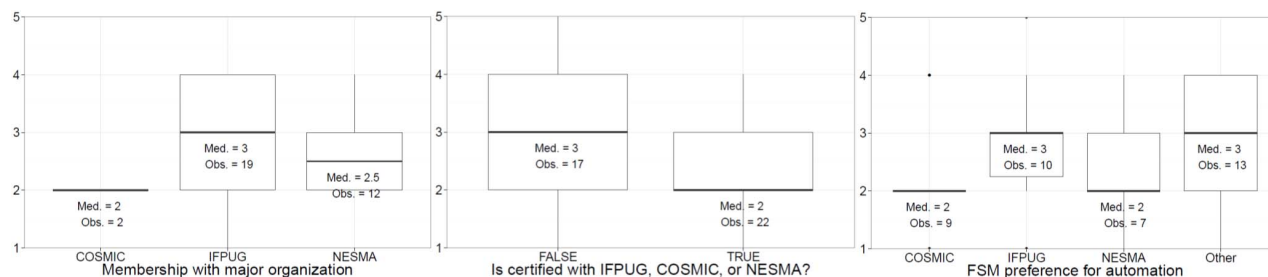


Figure 3. Boxplots on different sub-selections of respondents with regard to the question ‘Backfiring is a reliable measurement tool for conversion of Lines of Code data into Functional Size data’.

in fixed price projects” [R144]. “It is not suitable when it’s necessary to have a detailed count. And this detailed count will be used in the financial part of the project (like cost)” [R108]. Finally, it was mentioned that automated FSM is unsuitable for benchmarking purposes and for productivity analysis. “Automated FSM has no value for benchmarking. The result of the count is too much dependent on coding standards, architecture and other technological aspects that will vary over the companies” [R274]. “It will not be compatible with current IFPUG standards so separate benchmarking is required” [R231]. “Measuring the success of a project” [R253].

4.4 RQ4: Backfiring and FSM Tools

Both backfiring in the meaning of calculation of Lines-of-Code into Functional Size and calculation of Functional Size into Lines-of-Code are rated by a vast majority of the respondents as unreliable. Only 23% agree on the statements that say that backfiring is reliable. However, a high Coefficient of Variation for both statements indicate that a relatively large number of respondents agrees with these statement too, indicating different opinions. A strong positive correlation is found between both statements (see Table 3). Further analysis of the free format text revealed the following most mentioned reasons with regard to backfiring:

1. Unreliable due to high margin of error (32%).
2. Unreliable due to differences in programming styles, languages, architectures (28%).
3. Can be used within one domain (24%).
4. Can be used but is not reliable (16%).

Although many respondents do rate backfiring as unreliable, we notice that a relatively high variance score for both research questions indicate that respondents from different sub-selections do disagree on these statements (see Table 2). When looking at the boxplots in Figure 3 it shows that relatively large differences occur between all sub-selections, although the relatively low number of answer indicates low significance too. Apparently respondents that are not certified, and members of IFPUG or with a preference for IFPUG for automation do think easier on the reliability of backfiring than others. Respondents with a role other than business or IT, including respondents with a research background, are most condemned on backfiring: “Due to wide variation in completed sizes” [R099], and “Inaccurate, too much influence from the programming style” [R197].

In spite of the overall feeling of non-reliability of backfiring, it is used in practice, with mixed results: “We are doing this, but I think Automated Sizing using Cosmic would work better. More formal, comparable outside our own company” [R214]. “From experience

I now that coding standards and architecture variation will cause significant differences in conversion factors” [R284].

We see an interesting link between both statements on backfiring with the statement on ‘the tool(s) that I use for FSM satisfies my company’s needs’. A majority of respondents agrees with this: The Top-2-Box is 59%. In a way this surprises us because many commercial measurement tools that use FSM are based on backfiring (although, this is not mentioned by any of the respondents). Analysis of free format text revealed that among the most mentioned reasons why respondents are satisfied with their FSM tools are “The tool supports estimation based on historical data”, “Standardization, and combined with based on the OMG standard”, “Supports (faster) decision making”, and “They are reliable and efficient”.

The free format text reveals that relatively many respondents use self-made tools that support reporting on FSM and keeping track of data: “availability of historical data” [R269]. “Complexity based estimation based on historical references” [R095]. “We do have good tools for documenting the FSM and reuse them” [R242]. Others mention to be satisfied with commercial tools: “Commercial tool that is based on OMG AFP standards” [R089]. “I use the Starbuilder FP tools; it allows me to manage my projects in a professional manner.” [R039]. “The tool makes it possible to download a free viewer, so I can send anyone my FPA file and they can have a look at it. The only problem I see is that it is not supporting COSMIC.” [R202]. “MeterIT-Cosmic is COSMIC compliant” [R286]. “I use Price TruePlanning version 14.2 to primarily complete software cost estimates; works well with COSMIC” [R293]. “We use CAST Software on the delivered application to count functional size” [R069]. “We developed internal tools based on COCOMO and internal cost-driver models, and we use ISBSG, SEER, and QSM SLIM” [R053].

Respondents that rate not to be satisfied with their FSM tools mostly refer to the limited functionality and doubts on the quality of the outcomes: “Even when we have a tool, this let made a lot of decisions based in experience” [R167]. “Not completely because these tools concern only the base rules of measurement process” [R136]. “There are no good tools for FSM. Just methods. That’s not the same” [R212]. “There are no tools” [R213]. “I am not sure we can measure FP from source code. Experience done by CAST is not convincing.” [R324].

A remark is in place with regard to backfiring and tools. With regard to the relatively low number of respondents for these questions (39 for both questions related to backfiring, and 56 for tools), the outcomes with regard to these aspects must be looked upon with care. Although the survey results do not prove this, these outcomes

might imply that backfiring is not used much and that many companies do not use tools for FSM.

5. DISCUSSION

In this Section we discuss the results of our study and compare these with state of the art in industry, research, and education.

5.1 Threats to Validity

With regard to the extent to which the results of our study can be generalized to other situations and to other people, we argue that we encouraged a large variety of FSM specialists to answer the survey. By collaborating with the three major FSM associations we ensured a worldwide coverage of respondents from different backgrounds, as shown in Figure 1. However, we specifically addressed our survey to FSM specialists. Within this population our findings might be generalized. The outcomes however, cannot be generalized to people outside this group, such as for example decision makers and business executives responsible for IT investments and innovations.

5.2 Impact / Implications

Industry: Respondents that are for a major part from industry, indicate that automated FSM based on code should be an important tool mostly suited for baselining and benchmarking of software applications in maintenance and legacy environments. A majority of respondents sees COSMIC as most suited for this purpose. Based on the survey outcomes, we speculate that a solution for automated FSM that focusses on these requirements can help both FSM experts and decision makers. Besides that, we assume a need for such a solution in agile delivery environments, where speed of delivery of many subsequent iterations can be supported by automation of FSM based on code.

Research: Due to the assumed difficulties of automation of FSM based on code – the difference between a functional and a technical view, and the diversity in programming languages – we think that a focus within the research community on translation from functional counting rules towards technical programming code might be of importance. With regard to future work, an ‘OMG-like’, open-source approach focussing on the COSMIC method seems desired and interesting, where we assume that close cooperation with FSM communities will be valuable for translation towards industry.

Education: Looking at the fact that only 16% of the respondents is a starter in the field of FSM, while 41% has 10 years or more experience, we argue that FSM needs to be promoted in a better way among young IT professionals. Perhaps the FSM communities can play a role in this together with educational institutions.

6. CONCLUSIONS

A vast majority (87%) of the 336 FSM specialists that answered our survey considers FSM to be an important tool for decision making (RQ1). No indications are found that indicate any perceived impact of agile methodology on the difficulty of applying FSM (RQ2). 42% of the respondents says automated FSM is important, although many are uncertain (neutral) about this. A vast majority of respondents (50%) expects it to be difficult, while many are neutral on the question whether this idea will be difficult. 54% of the respondents think that automated FSM will help measurement specialists, while 44% thinks that it will help decision makers. The most preferred FSM method for automation is COSMIC (25%), followed by IFPUG (21%). Respondents think that automated FSM will be most suitable for baselining, benchmarking, and maintenance and legacy purposes (RQ3). Backfiring is perceived by a

majority of respondents as unreliable. 59% of the respondents is satisfied with the FSM tools they are currently using (RQ4).

ACKNOWLEDGMENTS

We sincerely thank all FSM enthusiast that spent time to provide us with valuable insights into their daily work. Once and for all their great cooperation shows that FSM is alive and kicking. Furthermore, we thank the boards of IFPUG, Nesma, and COSMIC for their inspiring effort in connecting us to their communities.

REFERENCES

- [1] A. F. Minkiewicz, “The Evolution of Software Size: A Search for Value,” *Software Engineering Technology*, vol. March/April, pp. 23-26, 2009.
- [2] C. Gencel and O. Demirors, “Functional Size Measurement Revisited,” *ACM Transactions on Software Engineering and Methodology*, vol. 17, no. 3, pp. 15:1-15:36, June 2008.
- [3] E. Ungan, O. Demirörs, Ö. Ö. Top and B. Özkan, “An Experimental Study on the Reliability of COSMIC Measurement Results,” *Software Process and Product Measurement*, no. Springer Berlin Heidelberg, pp. 321-336, 2009.
- [4] A. Živkovič, M. Heričko and T. Kralj, “Empirical assessment of methods for software size estimation,” *Informatica (Ljubljana)*, vol. 4, pp. 425-432, 2003.
- [5] Beck et al., “Manifesto for Agile Software Development,” 2012. [Online]. Available: www.agilemanifesto.org.
- [6] B. Meyer, *Agile!: The Good, the Hype and the Ugly*, Springer Science & Business Media, 2014.
- [7] A. Albrecht, “Measuring Application Development Productivity,” in *Joint Share Guide, and IBM Application Development Symposium 14-17 October 1979*, Monterey, California, 1979.
- [8] COSMIC, COSMIC-FFP: ISO/IEC 19761:2011 - Software engineering. A functional size measurement method, London: Common Software Measurement International Consortium (COSMIC), 2011.
- [9] IFPUG, IFPUG FSM Method: ISO/IEC 20926 - Software and systems engineering – Software measurement – IFPUG functional size measurement method, New York: International Function Point User Group (IFPUG), 2009.
- [10] UKSMA, Mk II Function Point Analysis: ISO/IEC 20968 - Software engineering – MI II Function Point Analysis – Counting Practices Manual, London: UK Software Metrics Association (UKSMA), 2002.
- [11] Nesma, Nesma functional size measurement method conform ISO/IEC 24570, version 2.1, Netherlands Software Measurement User Association (Nesma), 2005.
- [12] FiSMA, FiSMA FSM: ISO/IEC 29881 - Information technology – Software and systems engineering – FiSMA 1.1 functional size measurement method, Helsinki: Finnish Software Metrics User Association (FiSMA), 2010.
- [13] O. Mendes, A. Abran and P. Bourque, “Function Point Tool Market Survey,” *Software Engineering Management Laboratory*, Université du Québec à Montréal, 1996.

- [14] V. T. Ho and A. Abran, "A Framework for automatic function point counting from source code," in *International Workshop on Software Measurement (IWSM)*, 1999.
- [15] Object Management Group (OMG), "Automated Function Points (AFP)," Formal/2014-01-03 - Version 1.0, 2014.
- [16] R. Ellafi and R. Meli, "A Source Code Analysis-based Function Point Estimation Method integrated with a Logic Driven Estimation Method," in *SMEF*, 2006.
- [17] "Measuring Size & Productivity With CAST Automated Function Points," CAST Software, 2011.
- [18] L. Lavazza, "Automated Function Points: Critical Evaluation and Discussion," in *IEEE/ACM 6th International Workshop on Emerging Trends in Software Metrics (WETSoM)*, 2015.
- [19] A. Živkovič, I. Rozman and M. Heričko, "Automated software size estimation based on function points using UML models," *Information and Software Technology*, vol. 47, no. 13, pp. 881-890, 2005.
- [20] H. Diab, M. Frappier and R. St-Denis, "A formal definition of COSMIC-FFP for automated measurement of room specifications," in *Proc. 4th Eur. Conf. Software Measurement and ICT Control*, Heidelberg, 2001.
- [21] H. Diab, F. Koukane, M. Frappier and R. St-Denis, "µ c ROSE: automated measurement of COSMIC-FFP for Rational Rose RealTime," *Information and Software Technology*, vol. 47, no. 3, pp. 151-166, 2005.
- [22] S. Azzouz and A. Abran, "A proposed measurement role in the rational unified process and its implementation with ISO 19761: COSMIC-FFP," in *Software Measurement European Forum*, Rome, Italy, 2004.
- [23] B. Marín, O. Pastor and A. Abran, "Towards an accurate functional size measurement procedure for conceptual models in an MDA environment," *Data & Knowledge Engineering*, vol. 69, no. 5, pp. 472-490, 2010.
- [24] N. Condori-Fernández and Ó. Pastor, "Evaluating the productivity and reproducibility of a measurement procedure," *Advances in Conceptual Modeling-Theory and Practice*. Springer, pp. 352-361, 2006.
- [25] Z. Li, M. Nonaka, A. Kakurai and M. Azuma, "Measuring functional size of interactive software: a support system based on XForms-format user interface specifications," in *IEEE Third International Conference on Quality Software*, 2003.
- [26] D. Ceke and B. Milasinovic, "Automated web application functional size estimation based on a conceptual model," in *IEEE 2015 23rd International Conference on Software, Telecommunications and Computer Networks (SoftCOM)*, 2015.
- [27] R. Gonultas and A. Tarhan, "Run-Time Calculation of COSMIC Functional Size via Automatic Installment of Measurement Code into Java Business Applications," in *IEEE 41st Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*, 2015.
- [28] H. Soubra, L. Jacot and S. Lemaire, "Manual and Automated Functional Size Measurement of an Aerospace Realtime Embedded System: A Case Study based on SCADE and on COSMIC ISO 19761," 2015.
- [29] B. Marín, G. Giachetti and O. Past, "Measurement of functional size in conceptual models: A survey of measurement procedures based on COSMIC," *Software Process and Product Measurement*, no. Springer Berlin Heidelberg, pp. 170-183, 2008.
- [30] H. Soubra, A. Abran, S. Stern and A. Ramdan-Cherif, "Design of a Functional Size Measurement Procedure for Real-Time Embedded Software Requirements Expressed using the Simulink Model," in *IWSM Mensura*, 2011.
- [31] K. Lind and R. Heldal, "A model-based and automated approach to size estimation of embedded software components," *Model Driven Engineering Languages and Systems*, no. Springer Berlin Heidelberg, pp. 334-348, 2011.
- [32] H. Huijgens, M. Bruntink, A. v. Deursen, T. v. d. Storm and F. Voegelzang, "An Exploratory Study on Automated Derivation of Functional Size based on Code - Technical Report TUD-SERG-2016-007," Delft University of Technology, Delft, The Netherlands, 2015.