Thomas Fehlmann[1], Eberhard Kranich[2]

Do We Measure Functional Size? Or Do We Count? Functional Sizing in View of the Metrology Standards

Abstract

Functional Point counting has evolved into Functional Size Measurement. However, is it really a measurement? On the other hand, is this an imprecise change of name only? Measurements have a stray area, because no measurement instrument is completely exact. Counts in turn are exact. If a countable item does not go into the count, the count is imperfect but does not become a measurement.

In this paper, we present today's most popular counting methods and look in detail how they turn a function point count into functional size. We discuss SNAP, the *Software Non-functional Assessment Process* introduced by IFPUG that adds something like a quality size to software. We discuss the various other assessment processes that are likely to become important in the next few years, because they are urgently needed, such as security size assessments, privacy threat assessments and safety indices. All these are size assessments, which are quite different from functional size.

The aim of software metrics has long focused on predicting the cost of software development projects. For this, in the past, functional size was dominant. Today, functionality is available from cloud services against cost per use, while other aspects such as privacy, security and safety will become dominant concerns when designing and implementing software service.

This paper demonstrates how to use Six Sigma transfer functions for turning function point counts into measurements, be it functional, non-functional, safety, privacy, security or else.

Keywords: Function Points, Functional Size, Non-functional Size, Six Sigma Transfer Functions, software assessments, Metrology Standards, security, safety

1 Euro Project Office AG, Zeltweg 50, 8032 Zürich, Switzerland, thomas.fehlmann@e-p-o.com

2 Euro Project Office, Duisburg, Germany, eberhard.kranich@e-p-o.com

1. Metrology and Measurement Accuracy

Every measurement has a precision range expressing how reliable the measurement is. In contrast, a count does not have a range of dispersion. Thus, is functional sizing a measurement? Or rather a count, as expressed with the traditional term *Function Point Counting* that some years ago was used instead of functional sizing? How does functional sizing compare to the standards of metrology?

Abran presents in his book about software metrics and software metrology[3] a critique of the current approaches to functional size measurement. In fact, function point counts rather act as controls for functional size measurement. Transfer functions[4] can map a functional size count into some functional size measurement. A count – be it IFPUG, COSMIC, or else – is by itself not a measurement in the sense of the metrology standards. These standards are the *International Vocabulary of Metrology* (VIM)[5], and the *Guide to the Expression of Uncertainty in Measurement* (GUM)[6], set by the *Bureau International des Poids et Mesures* (BIPM). A measurement has its range of variation, indicating the measurement's precision. However, functional size has no stray area. Although it is precise in the sense[7] that quantity values obtained by replicating measurements agree, as long as the counter does not violate the counting rules, the variation remains unknown. Thus, as long as the sources of counting errors remain unexplored, it is unclear what a count measures.

Albrecht[8] positioned the function point counting method as a transfer function mapping EI, EO, EQ, ILF and EIF into something equivalent for business value. However, business value is difficult to measure[9]. Function point counting

---

3 A. Abran, *Software Metrics and Software Metrology*, I.C. Society, John Wiley and Sons, Hoboken, New Jersey 2010.

4 T.M. Fehlmann, E. Kranich, *Managing Complexity – Uncover the Mysteries with Six Sigma Transfer Functions*, Logos Verlag, Berlin 2016 (to appear).

5 ISO/IEC Guide 99:2007, *International Vocabulary of Metrology – Basic and General Concepts and Associated Terms (VIM)*, TC/SC: ISO/TMBG, Geneva, Switzerland 2007.

6 ISO/IEC CD Guide 98–3, *Evaluation of Measurement Data – Part 3: Guide to Uncertainty in Measurement (GUM)*, TC/SC: ISO/TMBG, Geneva, Switzerland 2015.

7 ISO/IEC Guide 99:2007, *International Vocabulary of Metrology – Basic and General Concepts and Associated Terms (VIM)*, TC/SC: ISO/TMBG, Geneva, Switzerland 2007, p. 25.

8 A.J. Albrecht, *Measuring Application Development Productivity*, Proceedings of the Joint SHARE, GUIDE, and IBM Application Development Symposium, October 14–17, Monterey, California 1979.

9 Z. Bakalova, *Towards Understanding the Value-Creation in Agile Projects*, vols. 13–288, CTIT Dissertation Series, Enschede 2014.

became popular because it proved to be equivalent to development effort under conditions that the ISBSG data collection practice describes[10].

On the other hand, COSMIC maps data movement counts into some kind of value created by integrating various knowledge gained from data sources and applications. This makes COSMIC attractive for measuring mobile apps and software services[11]. The creators of COSMIC looked at embedded systems that nowadays emerged into the *Internet of Things* (IoT). Data movements providing interconnectivity add value to today's web of things, including the investigation and prevention of defects, safety and security flaws created with software today.

1.1. What Is So Special about Functional Size Measurement?

Functional size measurement based on function points counting is different from what the VIM calls a measurement as neither software nor the underlying user requirements are the direct objects of the count. The objects are rather models of software. Such models exhibit certain aspects of software, and others remain hidden. As with a model of a house (Fig. 1), it is only possible to count what the model exhibits, e.g., doors, windows, roof windows, while other important aspects like heating, water pipes, isolation, etc. remain hidden. Most architects, therefore, use more than just one model to explain their building plans.

On the other hand, a model allows exact, digital counts. The question is whether the model is accurately representing the software under scrutiny. Thus, measurement accuracy based on counting model elements refers to the model validity in view of the measurement goals and not to the counting rules. Abran uses the term *Validation* of the model against *Verification* of the measurement context to discuss measurement accuracy[12]. Counts are something else than measurements.

Every function point counting method states, "defining the goal" of a count as its primary process step of the method. Building a model depends on the stated goal. In addition, the model might not be accurate with respect to this goal. The challenging problem is that the model accuracy is not expressible in the same dimensions as the measurement units themselves. It makes little sense to say that some COSMIC model has "maybe a few data movements more or less". In view of the IFPUG method, it is advisable to set all complexity scores to low,

---

10 *Practical Software Project Estimation*, 3rd edition, ed. P. Hill, McGraw-Hill, New York 2010.

11 A. Abran, *Software Metrics and Software Metrology*, I.C. Society, John Wiley and Sons, Hoboken, New Jersey 2010.

12 Ibidem, p.34.

then to high, in order to get something resembling a measurement range. Nevertheless, this is not sound from a metrological point of view, either; essentially this means counting an incomplete model, lacking information. This has nothing to do with a lack of precision.

Figure 1. House Model

Thus, measuring the model means counting. There is no precision range because the count is digital. Measuring characteristics of software or services should come with a precision indication. The precision depends on how well the model is able to reflect the true characteristics of software or services. For instance, counting one data movement to get an authentication from some service provider might be appropriate for cost estimation, if the functional user's viewpoint is to understand what happens. If the programmer needs to set up such an *Application Programming Interface* (API), he might see a bunch of data movements going forth and back, setting up a communication session and establishing a communication protocol. Thus, the measurement precision of the count depends on the measurement goal and the viewpoints of the functional users.

1.2. The Role of Transfer Functions in Measurements

It is obvious how to resolve that puzzle. We have the measurement goal, and we have counts. In order to get the measurement precision right, the measurement goal must exhibit a clear profile, which means the goal is measurable by the relative weights of the topics addressed. Such a profile is a *Goal Profile*; see for instance *Transfer Functions, Eigenvectors and QFD in Concert*[13]. It is recom-

[13] T.M. Fehlmann, E. Kranich, *Transfer Functions, Eigenvectors and QFD in Concert*, Proceedings of the ISQFD 2011, Stuttgart, Germany 2011.

mended to rely on more than one count alone. In the traditional application scenarios for functional sizing, the measurement goal is to correctly predict the cost of software development projects, see for instance the authors' IWSM paper of 2012[14]. Other goals could be predicting defect density as in the authors' IWSM paper of 2014[15].

In both cases, getting a goal profile is straightforward. The controls are the various counts applied to the project. Measuring the transfer function might be less straightforward, as the impact of each count on the measurement goals might either need consensus among experts (the QFD-way) or a regression analysis among similar projects previously conducted (the Six Sigma way). Data for such measurements might not be readily available.

1.3. Regularization

The distance between two measurement points plays a central role when comparing them. In mathematics, the Lp spaces are known as function spaces defined using a natural generalization of the distance norm for finite-dimensional vector spaces.

Say n is the dimension of the vector space; then, the p-norm definition is

$$\mathrm{Lp}(\langle x_1, x_2, \ldots, x_n \rangle) = \sqrt[p]{x_1^p + x_2^p + \ldots + x_n^p} \tag{1}$$

for $p = 1, 2, \ldots$ In measurement practice, $p = 1$ and $p = 2$ play the most prominent roles, representing distance in the L_1 and L_2 spaces. Statistics rely on the L_2 norm, predictive analysis as well. Economic models rely on the L_1 norm.

$$L_1(\langle x_1, x_2, \ldots, x_n \rangle) = \sqrt[1]{x_1^1 + x_2^1 + \ldots + x_n^1} = |x_1| + |x_2| + \ldots + |x_n| \tag{2}$$

$$L_2(\langle x_1, x_2, \ldots, x_n \rangle) = \sqrt[2]{x_1^2 + x_2^2 + \ldots + x_n^2} = \left\| \langle x_1, x_2, \ldots, x_n \rangle \right\| \tag{3}$$

For comparing vectors, the L_2 norm compares by the overall vector length, while the L_1 norm looks biased towards the maximum components value[16].

---

[14] T.M. Fehlmann, E. Kranich, *Quality of Estimations*, Proceedings of the IWSM/ Mensura, Assisi, Italy 2012.

[15] T.M. Fehlmann, E. Kranich, *Defect Density Measurements Using COSMIC – Experiences with Mobile Apps and Embedded Systems*, IWSM Mensura 2014, Rotterdam 2014.

[16] S. Bektas, Y. Sisman, *The Comparison of L1 and L2-norm Minimization Methods*, "International Journal of the Physical Sciences", September 18, 2010, vol. 5, no. 11, pp. 1721–1727;

Regularization, in mathematics and statistics and particularly in the fields of machine learning and inverse problems, refers to a process of introducing additional information in order to solve an ill-posed problem or to prevent over-fitting. In statistics and machine learning, regularization methods are used for model selection, in particular to prevent over-fitting by penalizing models with extreme parameter values. The most common variants in machine learning are L_1 and L_2 regularization, introducing a 'distance' between two events (in statistics) or two states (in machine learning).

Regularization for transfer functions uses the statistical way with the L_2 norm. Looking at the Taguchi loss function[17], it is apparent that Taguchi uses the L_2 norm.

The L_1 norm of any mathematical object is usually denoted by $|x|$ – the absolute value of a real number for instance as shown in equation (2) – while the common convention for the L_2 norm usually is $\|x\|$, as in equation (3).

1.4. Ratio Scales

Connected to the notion of regularization are *Ratio Type Scales*. The ratio type takes its name from the fact that measurement is the estimation of the ratio between a magnitude of a continuous quantity and a unit magnitude of the same kind. It has a zero point, and a unit value[18].

A ratio scale allows comparing different measurement results by their ratio. Saaty has introduced ratio scales for use with decision metrics in the *Analytic Hierarchy Process* (AHP)[19]; however, this lesson has not yet arrived fully in functional sizing, other approaches persist[20]. If certain criteria evaluate as either low, medium or high, each linked to a numerical value, without allowing intermediate

H.-H. Wu, *Using Target Costing Concept in Loss Function and Process Capability Indices to Set up Goal Control Limits*, "The International Journal for Advanced Manufacturing Technology" 2004, vol. 24, p. 206–213; T.M. Fehlmann, *Linear Algebra for QFD Combinators*, 9th International Symposium on Quality Function Deployment, Orlando 2003.

17 G. Taguchi, S. Chowhdury, Y. Wu, *Taguchi's Quality Engineering Handbook*, John Wiley & Sons, Hoboken 2005.

18 J. Michell, *Measurement Scales and Statistics: A Clash of Paradigms*, "Psychological Bulletin" 1986, vol. 3, p. 398–407.

19 T. Saaty, J. Alexander, *Conflict Resolution: The Analytic Hierarchy Process*, Praeger, Santa Barbara, CA, New York 1989.

20 A. Abran, *Software Metrics and Software Metrology*, I.C. Society, John Wiley and Sons, Hoboken, New Jersey 2010.

values, teams can more rapidly agree in interactive sessions. This is where the notion of "points" originates, as in function point counting.

Ratio scales in turn allow for comparing values. If on a scale 1, 3, 9 the nine is three times the three, not just something "high", it is possible to use arithmetics and finally statistical methods. Saaty was able to use the Eigenvector theory for decision-making[21]. That makes ratio scales superior as a measurement scale.

1.5. Separating Model Size Count and Functional Size Assessment

Counting the size of a model is straightforward. If the model consists of, say, n different element types, called *Model Dimension*, and c_i is the number of elements of type $i = 1, \ldots, n$ in the model, then the vector

$$\mathit{Model\ Count} = \langle c_1, c_2, \ldots, c_n \rangle \tag{4}$$

describes the model count with regard to the element types. Summing up this vector does not make any sense because the elements of the model may have different characteristics and yield different things depending on the question asked about the model.

Comparing model size is possible with the L_2 metric

$$\mathit{Model\ Size} = \left\| \langle c_1, c_2, \ldots, c_n \rangle \right\| = \sqrt{\sum_{i=1}^{n} c_i^2}\,, \tag{5}$$

which describes the Euclidean norm for a vector space of n dimensions.

2. Three Major Functional Sizing Methods

There are many methods to count function points; each starts with building a model of software or services and identifying model elements. Five of these methods are available as ISO/IEC standards, compliant to the standard ISO/IEC 14143–1:2007, which defines the concepts of *Functional Size Measurement* (FSM). Its concepts overcome the limitations of earlier methods by shifting the focus away from measuring how the software is implemented to measuring

[21] T.M. Fehlmann, E. Kranich, S. Schurr, *Analytic Hierarchy Process Made Easy – Report from the German AHP Working Group*, Stuttgart 2011.

size in terms of the functionality required by the user. The term for this user-required functionality is *Functional User Requirements* (FUR). The measurement unit for functional size is termed *Functional Size Unit* (FSU).

In view of the VIM[22] and the GUM[23], the criteria investigated for these three functional sizing methods are

Criterion 1) Does the method propose ratio scale metrics with a unit element?
Criterion 2) Does every measurement have a known and quantifiable variance?

Both criteria are necessary conditions for being a measurement. If one of these is violated, it still might be metrics, or a count; however, not a measurement.

2.1. ISO/IEC 20926:2009 IFPUG Function Point Counting

The FSU in ISO/IEC 20926 are the *IFPUG Function Point*, abbreviated IFP. This is the unit of measure for functional size within the international standard ISO/IEC 20926:2009; also referred as IFPUG 4.3.1. Complete details are available in the International Functional Size Unit Users Group (IFPUG) Counting Practices Manual 4.3.1, published by IFPUG January 2010[24]. The following five functional components of the software evaluate for the count according to ISO/IEC 20926 IFPUG rules based on user requirements:

- **Internal Logical File (ILF).** IFPUG 4.3.1: a user recognizable group of logically related data or control information maintained within the boundary of the application being measured.
- **External Interface File (EIF).** IFPUG 4.3.1: a user recognizable group of logically related data or control information referenced by the application being measured; however, maintained within the boundary of another application.
- **External Input (EI).** IFPUG 4.3.1: an elementary process that processes data, or control information sent from outside the boundary.
- **External Output (EO).** IFPUG 4.3.1: an elementary process that sends data or control information outside the boundary and includes additional processing logic beyond that of an External Inquiry.
- **External Inquiry (EQ).** IFPUG 4.3.1: an elementary process that sends data or control information outside the boundary.

---

22 ISO/IEC Guide 99:2007, *International Vocabulary of Metrology – Basic and General Concepts and Associated Terms (VIM)*, TC/SC: ISO/TMBG, Geneva, Switzerland 2007.

23 ISO/IEC CD Guide 98–3, *Evaluation of Measurement Data – Part 3: Guide to Uncertainty in Measurement (GUM)*, TC/SC: ISO/TMBG, Geneva, Switzerland 2015.

24 IFPUG Counting Practice Committee, *Function Point Counting Practices Manual – Version 4.3.1*, International Function Point User Group (IFPUG), Princeton Junction, NJ, 2010.

The model of ISO/IEC 20926 IFPUG has five constituent elements; thus, model counts are vectors in a five-dimensional vector space. According to equation (5), its model size is five. The arrows in Fig. 2 refer to *File Type Referenced* (FTR)[25]. They are not data movements, as in the functional size of an application is the sum of the sizes of its functional processes. The unit of measurement in COSMIC is equivalent to one single data movement type at the sub-process level. The COSMIC Function Point unit CFP defines a ratio scale and makes functional sizing available for analysis with statistical tools. The model size is four. This makes the COSMIC model attractive for statistical methods based on functional size in software.

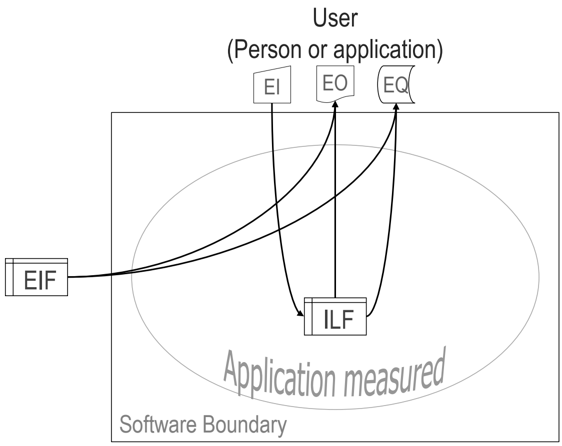

Figure 2. Functional Size Unit Counting Model

Source: the authors' own study.

Additionally, in the data functions the number of data fields (DET) and variant records (RET) are counted. The number of functional components, plus FTR, DET and RET would yield an interesting metric for software size[26], in line with Criterion 1).

However, functional size according to ISO/IEC 20926 IFPUG is defined in a way that makes the definition of the FSU a non-ratio scale type. There is neither a zero nor a unit size. After assigning complexity levels to each of the

25 Ibidem.

26 A. Dasgupta, C. Gencel, C. Symons, *A Process to Improve the Accuracy of MkII FP to COSMIC Size Conversions: Insights into the COSMIC Method Design Assumptions*, Software Measurement – IWSM Mensura, Kraków 2015.

functional components, based on FTR, DET and RET in the data functions, each functional component receives a fixed number of function points defined in a table. Intermediate values are not possible, making it difficult to apply statistical methods to an IFPUG count.

In practice, this shortcoming of the method is not important, as nobody supposes function point counts to be exact, because the underlying IFPUG model is not a full representation of the real software being measured. The IFPUG count aims at predicting the software development cost, which it did quite well for software projects of the past[27]. However, nobody can tell how much an IFPUG functional size count diverges and therefore, IFPUG counts are not measurements in the sense of the two criteria based on the VIM and the GUM.

2.2. ISO/IEC 19761 COSMIC

The ISO/IEC 19761 COSMIC measurement method originates from the perceived weakness of measuring real time software by the transaction–based ISO/IEC 20926 framework. Its first version was published in 1998, thus more than twenty years later than the IFPUG method, and is now available in the fourth updated version[28].

The COSMIC Functional Size Measurement Method. The principles behind COSMIC are:

- FUR generate *Functional Processes*. A functional process is "an elementary component of a set of FUR comprising a unique cohesive and independently executable set of data movements. It is triggered by one or more triggering events… it is complete when it has executed all that is required to be done in response to the triggering event"[29]. Triggering events occur outside the software boundary.
- Software manipulates pieces of information, designated as data groups, which consist of data attributes. **Fig. 3** depicts the data group flow.
- Functional processes involve sub-processes, concerned with movements – **E**ntries (E), e**X**its (X), **R**eads (R), and **W**rites (W) – and transformations of data groups.

[27] *Practical Software Project Estimation*, 3rd edition, ed. P. Hill, McGraw-Hill, New York 2010.

[28] COSMIC Measurement Practices Committee, *The COSMIC Functional Size Measurement Method – Version 4.0 – Measurement Manual*, The COSMIC Consortium, Montréal 2014.

[29] Ibidem.

- The functional size of a functional process is directly proportional to its number of data movements.

The functional size of an application is the sum of the sizes of its functional processes. The unit of measurement in COSMIC is equivalent to one single data movement type at the sub-process level. The COSMIC Function Point unit CFP defines a ratio scale and makes functional sizing available for analysis with statistical tools. The model size is four. This makes the COSMIC model attractive for statistical methods based on functional size in software.

Sizing an application with COSMIC involves the identification of functional processes, that of data groups, and that of data groups movements. There must be at least one **E**ntry, and one e**X**it or **W**rite. Each functional process involves at least two data movements. There is no upper limit on data movements; the smallest size for a functional process is two: an entry needed to trigger the functional process, and something happening such as an e**X**it or a **W**rite.

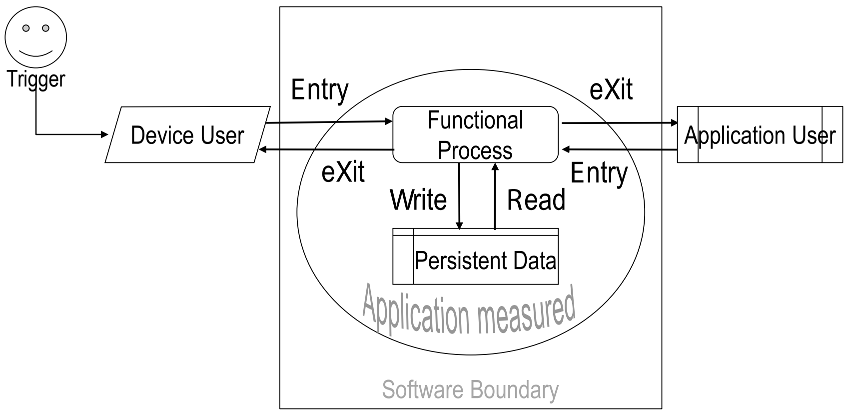

Figure 3. COSMIC Counting Model

Source: the authors' own study.

While ISO/IEC 19761 COSMIC uses a ratio scale, and thus meets Criterion 1), the level of granularity controls measurement accuracy but there is no metric for it. Model accuracy is not part of the ISO/IEC 19761 standard, thus violating Criterion 2). The variability of counts depends on the identification of data groups when moving data. ISO/IEC 19761 COSMIC is the only functional sizing standard without specific constructs aiming at predicting cost.

2.3. ISO/IEC 29881:2010 FiSMA

The method ISO/IEC 29881:2010 FiSMA 1.1 has been designed for those persons associated with the acquisition, development, use, support, maintenance, and audit of software. FiSMA 1.1 assesses FUR and measures functional size of a piece of software from the perspective of the users. Looking at the software architecture, the method identifies 28 different types of *Base Functional Components* (BFC); see Fig. 4. These BFC services consist of data movements, similar to COSMIC data movements as constituents of functional processes. The ISO/IEC 29881 FiSMA method is interesting because the model size is 28 and thus significantly higher than every other FSM.

A set of seven equations compute functional size and thus define how to transfer the model count into a functional size assessment. Each model element receives a weight based on their type: for instance, with interactive input, equation (6) is valid for the BFC service (i1) based on the number of output data presented (n), of writing references (w) and of reading references (r). The variable $m = 1, 2, 3$ stands for one, two or three possible interaction types of CrUD (Create–Update–Delete), and *FFP* for *FiSMA Function Points*:

$$FFP = m * \left(0.2 + \frac{n}{5} + \frac{w}{1.5} + \frac{r}{2}\right) \tag{6}$$

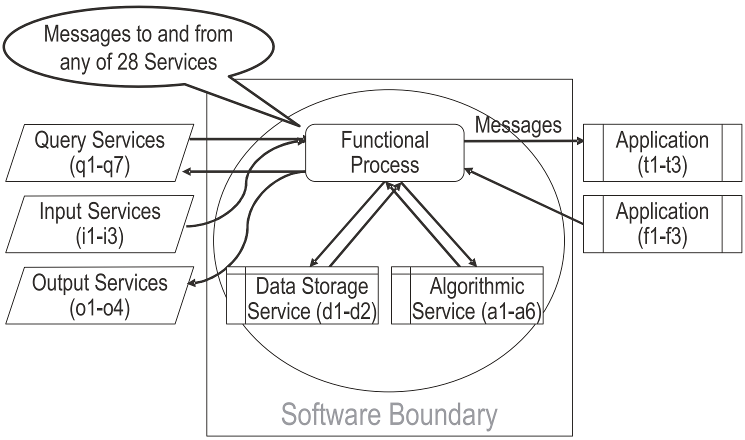

Figure 4. ISO/IEC 29881 FiSMA Function Point Model

Source: the authors' own study.

Combining similar formulas for all seven BFC groups yields the total FFP. According to metrology standards, this is impossible because adding a different type of counts makes no sense mathematically. One might argue that the cryptic factors – here in (6): 0.2 – make the different types compatible; however, the ISO/IEC 29881 FiSMA standard does not explain how.

The FiSMA functional size count is a regular ratio scale type but not a measurement. It meets Criterion 1), however, with the particularity that a measurement unit is not linked to any observable entity. The unit depends on the linear constants used to make the 28 BFC comparable to each other. In equations (6), this is the constant 0.2. These constants aim at predicting cost. Criterion 2) is violated.

3. Solution Approaches

In the past, the problem with Criterion 2) has been approached by different solution approaches.

3.1. Uncertainty

Adding the notion of uncertainty that pertains to the model is certainly a way to go. Santillo[30] and Fehlmann and Santillo[31] presented various ideas based on analyzing the measurement process and looking for variations introduced with assumptions about the model. Another approach to assess uncertainty is by looking at requirements elicitation and its evolution. Usually, when conducting a software project, requirements are unclear at the beginning and evolve with the increasing knowledge about the topic, see Santillo[32]. With respect to FUR, the process of requirements elicitation is measurable by its speed and the requirements growth rate. With new requirements, the software model grows as well, and this speed can be taken as a measure for the accuracy of the model, and thus of the functional size count based on this model.

---

30 L. Santillo, *Error Propagation in Software Measurement and Estimation*, IWSM/MetriKon 2006, Potsdam 2006.

31 T.M. Fehlmann, L. Santillo, *Uncertainty of Software Requirements*, Proceedings of the 4th Software Measurement European Forum, Rome 2007.

32 L. Santillo, *Early and Quick COSMIC FFP Analysis Using Analytic Hierarchy Process*, in: *COSMIC Function Points – Theory and Advanced Practices*, eds. R. Dumke, A. Abran, Boca Raton, FL, CRC Press 2011, pp. 176–191.

3.2. Transfer Functions

The problem with such an approach is that if a functional size measurement varies over time, this is not characteristic for the measurement accuracy itself. It is rather because the thing being measured grows. Thus, transfer functions are a better candidate for assessing the measurement process for its accuracy. Transfer functions relate to the goal of the measurement process, mapping the size controls to the purpose of measurement. For instance, predicting project cost was always a major reason for function point counting. Such transfer functions map a number of controls – functional size and various cost drivers such as team size, time constraints, and technology – onto cost predictions.

Today's software projects are less dependent on functional size than on other cost drivers, such as security, safety, or collaboration requirements. Most projects do no longer rely on bespoken software only; they combine existing services and provide an interface to the user that combines various knowledge sources to provide excellent service; see the authors' paper at the IWSM 2012[33].

It is possible to pursue other measurement goals based on a functional count, yielding different measurement accuracy depending on the goal of measurement. This observation is another argument that counts are not measurements; if the same count can yield for instance a functional size measurement and a defect density measurement, the underlying count cannot be a measurement in its own right.

3.3. Measuring Functional Size

If the measurement goal is measuring functional size, the question arises which method to choose. Each model has its specific advantages[34]. So, why not use more than one model? Every architect in the building industry does this. ICT managers in turn seem not inclined to do so; sometimes, they try it even without assessing functional size. The notion of functional size benefits from combining several methods according to a weight vector that expresses the suitability of each count towards the measurement goal.

---

[33] T.M. Fehlmann, E. Kranich, *Quality of Estimations*, Proceedings of the IWSM / Mensura, Assisi, Italy 2012.

[34] T.M. Fehlmann, *When Use COSMIC FFP? When Use IFPUG FPA? A Six Sigma View*, in: *COSMIC Function Points – Theory and Advanced Practices*, eds. R. Dumke, A. Abran, Boca Raton, FL, CRC Press 2011, pp. 260–274.

To show how this works is the aim of the last chapter of this paper. Note that the same mechanism works well for almost any of possible measurement goals.

4. A Sample Functional Size Measurement

This example uses four different approaches for counting functional size in a project:

- An IFPUG count, to get the number of transactions and interfaces right;
- A COSMIC count, for understanding the data communication architecture;
- A suitable non-functional sizing count, e.g., SNAP[35] or the Buglione-Trudel Matrix, see Fehlmann & Kranich[36];
- An estimate for sizing legal constraints. *Legal Size* might be a metric based on counting regulations and legal constraints, with which the software has to be compliant.

Fig. 5 reflects a project having the following three measurement goals.

	Measurement Goals Topics	Attributes			Profile
y1	Predict Cost of Functionality	*Boundary right*	*Right model*	*Complete Coverage*	*0.57*
y2	Estimate Quality Effort	*Required quality*	*Legal constraints*	*Stakeholder value*	*0.77*
y3	Predict Overhead Cost	*Project Management*	*Stakeholder Management*	*Project Marketing*	*0.28*

Figure 5. Measurement Goals for a Sample Project

Source: the authors' own study.

The transfer function might look as shown in Fig. 6 below. The estimator might have used several functional sizing methods, wondering which one deserves confidence.

The level of confidence regarding the measurement goals depends on a project and may vary. The QFD matrix cells contain the level of confidence expressed by the estimator as correlation values, similar to using QFD for early project estimation. The measurement precision is the convergence gap, here 0.05, or 5% in relation to the unit length of the profiles. The impact measurements in this

[35] IFPUG Non-Functional Sizing Standards Committee, *Software Non-functional Assessment Process (SNAP) – Assessment Practices Manual*, International Function Point Users Group (IFPUG), Princeton Junction, April 2013.

[36] T.M. Fehlmann, E. Kranich, *Early Software Project Estimation the Six Sigma Way*, "Lecture Notes in Business Information Processing" 2014, vol. 199, pp. 193–208.

case are expert judgments the QFD way. It also yields the solution profile for counts, indicating how much weight each count has concerning the measurement goals, in this case a consolidated cost estimation for the whole project.

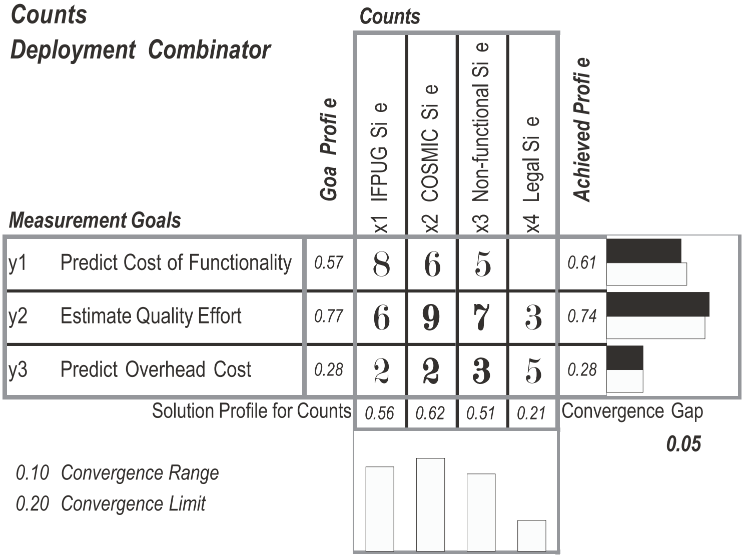

Figure 6. Transfer Function for the Measurement Accuracy

Source: the authors' own study.

Fig. 7 shows the estimation controls resulting from the transfer function. The impact measurements in this case are expert judgments in the QFD way. The solution profile for counts indicates how much weight each count has concerning the measurement goals. In this case, it refers to the consolidated cost estimation for the whole project. Thus, transfer functions make functional size counts compliant with the metrology standards. The convergence gap reflects measurement precision with regard to the chosen measurement goals. The solution weights profile (Fig. 7) shows the overall importance, or contribution, that each of the counts adds to the measurement goals. AHP[37] is the method of choice for determining importance. Thus, measurement precision is not with regard to the count but with regard to the measurement goals, in accordance

37 T.M. Fehlmann, E. Kranich, *Managing Complexity – Uncover the Mysteries with Six Sigma Transfer Functions*, Logos Verlag, Berlin 2016 (to appear); T. Saaty, J. Alexander, *Conflict Resolution: The Analytic Hierarchy Process*, Praeger, Santa Barbara, CA, New York 1989.

with the standards for measurement, the VIM and the GUM. Each count contributes to measurement precision in some specific way.

	Counts Topics	Attributes		Priority
x1	IFPUG Size	*ISO/IEC 20926*	*Transactions count*	*0.56*
x2	COSMIC Size	*ISO/IEC 19761*	*Count functional processes*	*0.62*
x3	Non-functional Size	*SNAP*	*Buglione-Trudel-Matrix*	*0.51*
x4	Legal Size	*Applicable Laws*	*Site Constraints*	*0.21*

Figure 7. Contribution Profile for the Various Counts Applied

Source: the authors' own study.

5. More Possible Assessment Methods

Adhering to the standards for measurement opens a wide range of possible measurements based on function point counts. In view of the role of sizing for other measurement purposes than effort estimation, why not make the complexity measurement just another assessment, probably in view of effort estimation? The Software Non-functional Assessment Process (SNAP) already works that way[38], and the need for something like a *Software Security Assessment Method* (SSecAM) or a *Software Safety Assessment Method* (SSafAM) is already emerging. This means that both the IFPUG and the COSMIC model, as well as FiSMA, and others possibly as well, could serve for measuring various aspects of software. A count makes measurements comparable; the measurements themselves depend on the accuracy of assessing each model element by the target criteria.

6. Conclusion

Functional sizing according to one of the ISO standards is a count of model elements, be it COSMIC functional processes, IFPUG transactions or FiSMA base functional components. The transfer function provides the expected accuracy,

[38] IFPUG Non-Functional Sizing Standards Committee, *Software Non-functional Assessment Process (SNAP) – Assessment Practices Manual*, International Function Point Users Group (IFPUG), Princeton Junction, April 2013.

based on the confidence assessment by the estimator. Thus, functional sizing yields a measure with the confidence interval.

With ISO/IEC 20926 IFPUG, split application counts do not easily add up. The IFPUG scale is not a ratio scale. This affects the suitability for counting small mobile apps that together constitute a system of interrelated applications. However, the IFPUG method identifies valuable components in its model, such as FTR, RET and DET, that could be used to define a metrics conformant with the metrology standards, suitable for counting today's mobile apps. It would be highly welcomed if the IFPUG community comes up with a new measurement scale conformant with both the metrology standards and the traditional low/medium/high complexity scale. Functional size measurement compliant with the metrology standards, the VIM and the GUM, has the potential to bring software metrics and measurement to the mainstream. Moreover, it would be an honor for Thomas Saaty who has propagated this for almost thirty years by now. It is an important step towards better managing the complexity of software development and service deployment.

References

Abran A., *Software Metrics and Software Metrology*, I.C. Society, John Wiley and Sons, Hoboken, New Jersey 2010.

Albrecht A.J., *Measuring Application Development Productivity*, Proceedings of the Joint SHARE, GUIDE, and IBM Application Development Symposium, October 14–17, Monterey, California 1979.

Bakalova Z., *Towards Understanding the Value-Creation in Agile Projects*, vols. 13–288, CTIT Dissertation Series, Enschede 2014.

Bektas S., Sisman Y., *The Comparison of L1 and L2-norm Minimization Methods*, "International Journal of the Physical Sciences", September 18, 2010, vol. 5, no. 11, pp. 1721–1727.

COSMIC Measurement Practices Committee, *The COSMIC Functional Size Measurement Method – Version 4.0 – Measurement Manual*, The COSMIC Consortium, Montréal 2014.

Dasgupta A., Gencel C., Symons C., *A Process to Improve the Accuracy of MkII FP to COSMIC Size Conversions: Insights into the COSMIC Method Design Assumptions*, Software Measurement – IWSM Mensura, Kraków 2015.

Fehlmann T.M., Kranich E., *Managing Complexity – Uncover the Mysteries with Six Sigma Transfer Functions*, Logos Verlag, Berlin 2016 (to appear).

Fehlmann T.M., Kranich E., *Transfer Functions, Eigenvectors and QFD in Concert*, Proceedings of the ISQFD 2011, Stuttgart, Germany 2011.

Fehlmann T.M., Kranich E., *Quality of Estimations*, Proceedings of the IWSM/ Mensura, Assisi, Italy 2012.

Fehlmann T.M., Kranich E., *Defect Density Measurements Using COSMIC – Experiences with Mobile Apps and Embedded Systems*, IWSM Mensura 2014, Rotterdam 2014.

Fehlmann T.M., *Linear Algebra for QFD Combinators*, 9th International Symposium on Quality Function Deployment, Orlando 2003.

Fehlmann T.M., Kranich E., Schurr S., *Analytic Hierarchy Process Made Easy – Report from the German AHP Working Group*, Stuttgart 2011.

Fehlmann T.M., *When Use COSMIC FFP? When Use IFPUG FPA? A Six Sigma View*, in: *COSMIC Function Points – Theory and Advanced Practices*, eds. R. Dumke, A. Abran, Boca Raton, FL, CRC Press 2011, pp. 260–274.

Fehlmann T.M., Kranich E., *Early Software Project Estimation the Six Sigma Way*, "Lecture Notes in Business Information Processing" 2014, vol. 199, pp. 193–208.

Fehlmann T.M., Santillo L., *Uncertainty of Software Requirements*, Proceedings of the 4th Software Measurement European Forum, Rome 2007.

IFPUG Counting Practice Committee, *Function Point Counting Practices Manual – Version 4.3.1*, International Function Point User Group (IFPUG), Princeton Junction, NJ 2010.

IFPUG Non-Functional Sizing Standards Committee, *Software Non-functional Assessment Process (SNAP) – Assessment Practices Manual*, International Function Point Users Group (IFPUG), Princeton Junction, April 2013.

ISO/IEC Guide 99:2007, *International Vocabulary of Metrology – Basic and General Concepts and Associated Terms (VIM)*, TC/SC: ISO/TMBG, Geneva, Switzerland 2007.

ISO/IEC CD Guide 98–3, *Evaluation of Measurement Data – Part 3: Guide to Uncertainty in Measurement (GUM)*, TC/SC: ISO/TMBG, Geneva, Switzerland 2015.

Michell J., *Measurement Scales and Statistics: A Clash of Paradigms*, "Psychological Bulletin" 1986, vol. 3, p. 398–407.

Practical Software Project Estimation, 3 rd edition, ed. P. Hill, McGraw-Hill, New York 2010.

Saaty T., Alexander J., *Conflict Resolution: The Analytic Hierarchy Process*, Praeger, Santa Barbara, CA, New York 1989.

Santillo L., *Error Propagation in Software Measurement and Estimation*, IWSM/ MetriKon 2006, Potsdam 2006.

Santillo L., *Early and Quick COSMIC FFP Analysis Using Analytic Hierarchy Process*, in: *COSMIC Function Points – Theory and Advanced Practices*, eds. R. Dumke, A. Abran, Boca Raton, FL, CRC Press 2011, pp. 176–191.

Taguchi G., Chowhdury S., Wu Y., *Taguchi's Quality Engineering Handbook*, John Wiley and Sons Hoboken 2005.

Wu H.-H., *Using Target Costing Concept in Loss Function and Process Capability Indices to Set up Goal Control Limits*, "The International Journal for Advanced Manufacturing Technology" 2004, vol. 24, p. 206–213.