Function Points: A Study of Their Measurement Processes and Scale Transformations

Alain Abran
Université du Québec à Montréal, Montréal, Québec, Canada

Pierre N. Robillard
École Polytechnique de Montréal, Mont&al, Québec, Canada

Function point metrics were initially designed through expert judgments. The underlying measurement model has not been clearly stated, and this has generated some confusion as to the true nature of these metrics and their usefulness in fields other than their initial Management Information System domain. When viewed without reference to implicit models hidden in the expert judgments, function points constitute a potpourri of measurement scales. This suggests that each step could represent a transcend the measurement scales and maintain or improve the desired relationship with development effort.

1. EVOLUTION OF FUNCTION POINTS

Function point metrics, developed by Allan Albrecht of IBM, were first published in 1979, and, in 1984, the International Function Point Users Group (IFPUG) was set up to clarify the rules, set standards, and promote their use and evolution. Function point metrics provide a standardized method for measuring the various functions of a software application. Function point metrics measure functionality from the user's point of view, that is, on the basis of what the user requests and receives in return. Over the years, various improvements have been made to the 1979 initial description, and successive versions have been published (Table 1). The first three versions addressed the structure of these metrics, whereas the three IFPUG versions addressed the clarification of the rules and guidelines.

The Albrecht 79 model (Albrecht, 1979) had four function types and one set of weights (Table 2, left), together with 10 general system characteristics (GSCs) for a maximum value adjustment factor (VAF) of $\pm 25\%$ (Table 3, left). The Albrecht 83 model (Albrecht and Gaffney, 1983) was expanded to five function types, three sets of weights (Table 2, right) and 14 GSC, for a maximum adjustment of -35% (Table 3, column 2).

An example of a project count using the Albrecht 83 version is presented in Table 4. The application measured in the example contains three internal files, zero external file, two inputs, two outputs, and five inquiries, all of average complexity, for a total of 68 unadjusted function points (UFP); in addition, the GSC 1-11 have no influence, whereas 12-14 are rated as significant, for a total degree of influence (TDI) of 12. These two results (UFP and TDI), combined into the total value adjustment formula ($UFP * [0.65 + 0.1 * TDI]$), produce 52 adjusted function points.

The GUIDE 85 version introduced a new dimension to function points through a set of rules for the functional complexity rating (low, average, and high) of the five function types: the function types were decomposed into three types of primary components (Table 5), and two-dimensional matrices with predetermined ranges of values were used for rating purposes, thus allowing consistent rating across individuals and organizations. Table 6 presents the

Address correspondence to Pierre N. Robillard, École Polytechnique de Montréal, P.O. Box 6079, Sta. A, Montréal, Québec H3C 3A7, Canada.

Table 1. List of Function Point Versions

Version	Reference
Albrecht 79	Albrecht, 1979
Albrecht 83	Albrecht and Gaffney, 1983
GUIDE 85	GUIDE, 1985
IFPUG 86	International Function Point Users Group, 1986
IFPUG 88	International Function Point Users Group, 1988
IFPUG 90	International Function Point Users Group, 1990

two-dimensional matrix designed for the datatype functions (internal logical files and external interface files). For example, a file with 2.5 data element types (DETs) and seven record types (RETs) would be rated as a file of high functional complexity.

The three subsequent versions published by IFPUG have provided further clarification of the rules, guidelines, and criteria, but they have not introduced any change to the structure of function point methodology itself. The current official version, IFPUG 90 (International Function Point Users Group, 1990)[1], still uses the Albrecht 83 function types and weights, as well as the GUIDE 85 decision matrices. Throughout this article, only the official IFPUG 90 version of function points terminology and rules will be used.

Function point metrics are derived from a set of steps, rules, and formulas: they are an algorithmic metrics, and therefore have the problems inherent in any algorithmic (or synthetic) metrics system:

- Algorithmic metrics are difficult to interpret. The reasons for the assignments of specific values (weights) are not clear.
- The value of the output of the formula is useful only if the formula is based on a solid theory, such as physics, but this not the case with these software metrics (Conderoy et al., 1989).

The next section explores the ambiguity of the function points definition and identifies the research issue from a measurement perspective. To address it, the function point measurement process is modeled in section 3, and the measurement scales used in function points are identified and discussed in section 4, together with related measurement issues. Finally, section 5 explores some of the issues related to the initial mapping space of function points.

2. AMBIGUITY OF FUNCTION POINTS DEFINITION

Most publications on function points have addressed issues extraneous to its structure, such as

- Comparisons with other software metrics based on lines of code, such as Halstead (Albrecht and Gaffney, 1983)
- Accuracy of estimates (Kemerer, 1987)
- Interrater reliability* (Kemerer, 1990; Low and Ross, 1990; Rudolph, 1983, 1989)
- Productivity analysis (Kemerer, 1987; Behrens, 1989; Emrick, 1988a, 1988b; Gaffney, 1986)

Only a few authors have reviewed function point methodology and identified some of its weaknesses in such areas as domain of applicability, structure of primary components, and impact of the value adjustment factor (Desharais, 1988; Benyahia et al., 1990; Jones, 1988b; Symons, 1988). However, none has formally addressed the measurement issue per se.

The function point definition itself, for example, has not been clarified, and has generated some confusion among both practitioners and academics, as illustrated in Table 7: does it measure size, productivity, complexity, functionality, or user deliverables? This ambiguity can be traced back to Albrecht's (1979) initial definition: "This gives a dimensionless number defined in function points

[1]IFPUG 90 refers to the latest major release of the IFPUG Counting Practice Manual (Version 3.0); since then, only minor changes to the counting rules and guidelines have been approved by IFPUG, the latest one being the September 1993 version 3.4.

[2]Interrater reliability is consistency of the function points results when different individuals measure the same software.

Table 2. Function Point Weights (Albrecht 79 and 83)

Albrecht 79		Albrecht 83			
Function Types	Weights	Function Types	LOW	Average	High
Files	10	Internal logical files	7	10	15
		External interface files	5	7	10
Inputs	4	External inputs	3	4	6
outputs	5	External outputs	4	5	7
Inquiries	4	External inquiries	3	4	6

Table 3. Function Point Adjustment Factors

General System Characteristics (GSC)		
Albrecht 79	Albrecht 83	Degrees of Influence
1. Backup	1. Reusability	0. None
2. Data communications	2. Data communications	1. Incidental
3. Distributed processing	3. Distributed processing	2. Moderate
4. Performance issues	4. Performance issues	3. Average
5. Heavily used configuration	5. Heavily used configuration	4. Significant
6. Online data entry	6. Online data entry	5. Essential
7. Conversational data entry	7. Conversation data entry	
8. Online update of master files	8. Online update of master files	
9. Complex functions	9. Complex functions	
10. Internal processing complex	10. Internal processing complex	
	11. Installation ease	
	12. Operational ease	
	13. Multiple sites	
	14. Facilitate change	
Value adjustment factor = (0.75 ± 25% max)	Value adjustment factor = (0.65 ± 35% max)	

which we have found to be an effective relative measure of function value delivered to our customer."

But what is a metric if it is only a number (Eijogu, 1990)? Measures must be associated with a modeling process, and, once a model is defined, measures are required to verify whether the model performs as projected. Numbers without a reference system and without a model are useless and cannot be used to analyze and derive information. "The goal should be a set of measures that can be justified theoretically, that can be used with confidence by both programmers and project managers" (Schen et al., 1983). Measures must also be based on the mathematical discipline of measurement theory. A measure must always represent a mapping to a specific model: an attribute can be measured if there is a mapping from an empirical relation system into a numerical relation system (Basili and Musa, 1991).

The research issue can be stated as follows: Where do function points stand with respect to measurement systems? The motivation for this research was to assess whether function point metrics yield a dimensionless number or a multidimensional index, and whether they have the basic characteristics and structural strength necessary to become a lasting reference measurement system. The research objective is the identification and analysis of the measurement system embedded within function point metrics. The research purpose is to evaluate the function points measurement method, and the perspective taken is that of the researcher: an analysis of the principles and mathematical foundations of these metrics. The domain of this analysis is limited to the identification of this issue, as opposed to the issue of its usefulness in productivity analysis and cost-estimation models.

Two investigation topics were defined as follows:

1. What are the formal measurement processes embedded within the function point metrics?
2. Which types of measurement scale (nominal, ordinal, interval, ratio, and absolute) are used in the various steps and measurement processes of function points?

3. FUNCTION POINT MEASUREMENT MODEL

Function point metrics have been modeled from a measurement perspective (Abran and Robillard,

Table 4. Example of a Function Point Count (Albrecht 83)

Function Types	Functions * N Weights	GSC and Degree of Influence
Internal Files	3 * 10 = 30	GSCs 1–11 = 0
External Files	0 * 7 = 0	
Inputs	2 * 4 = 8	GSC 12 = 4
outputs	2 * 5 = 10	GSC 13 = 4
Inquires	5 * 4 = 20	GSC 14 = 4
Total	UFP = 68	TDI = 12
AFP = 68 * (0.65 + 0.01 * 12) = 52		

Table 5. Types of Primary Components (Guide 85)

Abbreviation	Description
DET	Data element types
RET	Record types
FTR	File types referenced

1991), as illustrated in Figure 1. The function points are obtained through measurement of the application from two distinct perspectives.

- The functional size, calculated through the measurement of each individual function. This will be referred to as the functional size measurement process.
- The value adjustment factor, calculated through the measurement of the application as a whole. The objective of this factor is to assess the general functionality of the application. This will be referred to as the adjustment measurement process.

The value adjustment factor then adjusts the functional size by a maximum adjustment of $\pm 35\%$ to produce the adjusted function points (AFP). This is referred to in Figure 1 as the function points measurement model.

Functional Size

The functional size measurement process itself consists of two processes, the data measurement process for internal and external files and the transaction measurement process for inputs, outputs, and inquiries. These data and transaction measurements are themselves complex processes. For example, the data measurement process is further decomposed into five steps (Figure 2).

F1. The firs step takes as input the application/project documentation and, from the user-identifiable groups of logically related data, produces as output the list of logical files.[3] This step is based on an implicit model of the structure of data (Figure 3). Note that although this model is described in the methodology in terms of rules and guidelines, it is not based on a software engineering theory, and its basic relationships have not been formally analyzed.

F2. The second step analyzes the border between the application/project being measured and either external applications or the user domain in order to classify the logical files into internal or external files. The output of this process consists of sublists of internal logical files and external logical files.

F3. The third step counts the actual number of DETs[4] and RET[5] within each logical file type referenced (FTR).[6] Here, for the sake of clarity, the term *data elements* refers to the DETs, and record refers to the RETs, as defined in IFPUG (1990).

F4. The fourth step applies the data algorithm with the following five inputs: sublists of files, DET counts, RET counts, data functional complexity decision matrix (referred to as data complexity formula in Figure 2), and weights table. The output of this process is a list of points for all logical files.

F5. The last data measurement step consists of the addition of all points from step F4 to produce the unadjusted data points.

The transaction measurement process shown in Figure 1 is similarly decomposed, in this case into six steps (Figure 4).

T1. The first step takes as input the application/project documentation to identify the user's visible functions. This produces the lists of external inputs, external outputs, and external inquiries. Here, for the sake of clarity, these will be referred to only as inputs, outputs, and inquiries.

T2. The second step takes these lists of functions and the functions/transactions model to determine the list of countable transactions.

T3. In the third step, the logical files and the data identified in F2 and F3 are applied to the list of

Table 6. Decision Matrix Structure (GUIDE 85)

Data-Type Functions

	DETs		
RETs	1-19	20-50	≥ 51
1	Low	Low	Average
2-5	LOW	Average	High
≥ 6	Average	High	High

[4] **DET**: a unique occurrence of data, also referred to as a data element, variable, or field (IFPUG 1990).

[5] **RET**: a unique record format within an internal logical tile or external interface (IFPUG 1990).

[6] **FTR**: the number of internal logical files or external interfaces read, created, or updated by a function type (IFPUG 1990).

Table 7. Overview of Function Point Interpretation

Interpretation	Descriptions and Authors
Size	A size metric, Wrigley, 1988
	A surrogate metric for size, Cowderoy et al., 1989
Productivity	A product size measure to define a productivity measure, Cowderoy et at., 1989
	A productivity measure defined as the weighted total of functional units produced, Conte et al., 1986
	An economic productivity unit; Jones, 1988c
Complexity	A complexity measure of projects, Martin, 1991
Functionality	The functionality or utility of a program Pressman, 1987
	An indirect measure of functionality created by a software application; Pressman, 1988
User Deliverables	A numeric value assignment to user deliverables; Johnson, 1991
Dimensionless number	The calculation results do not have direct meaning, interpretation or physical analogy; Pfleeger 1989
	A dimensionless number. An abstract and synthetic number, but a useful and practical performance indicator.
	An artificial metric similar to the notion of the Dow Jones industrial average; Jones 1988a
Multidimensions	A parameter to quantity overall behavior; Dunn 1990

transactions to determine two sets of counts: the number of file types referenced (FTR) and the number of DETs referenced for each transaction.

T4A. The fourth step applies the transaction algorithm with the following five inputs: the list of transactions, file counts, data counts, transaction functional complexity decision matrix (referred to as transaction complexity formula in Figure 4), and transaction weight table. The output of this process is a preliminary list of points for all transactions.

T4B. The second phase of this step selects, for each inquiry function, the greater of the transaction input or output points, and eliminates the lesser points to produce the final list of transaction points.

T5. The last transaction measurement step is the addition of all transaction points to produce the number of unadjusted transaction points.

The addition of the unadjusted data points in F5 to the unadjusted transactions points in T5 produces the total unadjusted function points and gives the functional size (Figure 1) of the application.

Value Adjustment Factor

The adjustment measurement process, which produces the value adjustment factor, is similarly decomposed into five steps (Figure 5).

V1. The first step takes as input the application/ project documentation and the criteria defini-

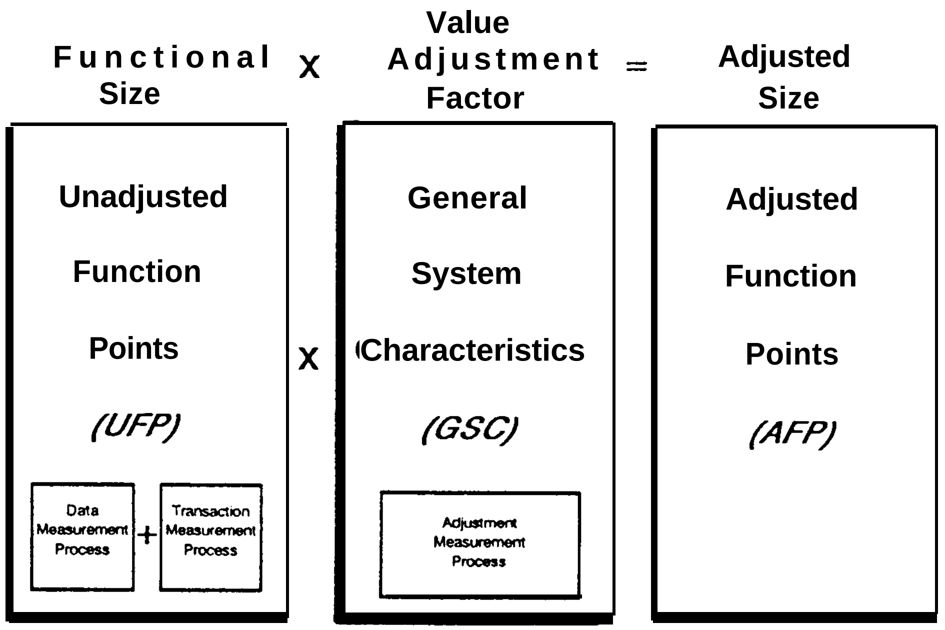
Functional Size
X
Value Adjustment Factor
=
Adjusted Size
Unadjusted Function Points (UFP)
X
General System Characteristics (GSC)
Adjusted Function Points (AFP)
Data Measurement Process
+
Transaction Measurement Process
Adjustment Measurement Process

Figure 1. Function points measurement model.

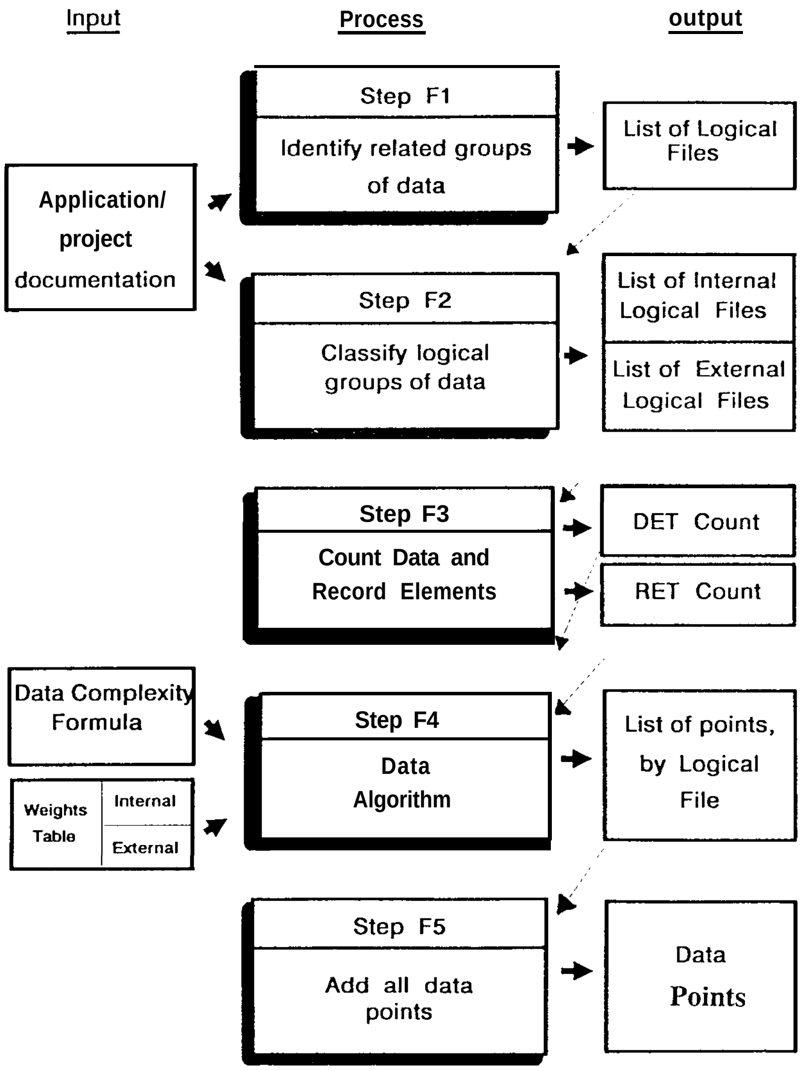
Input
Process
output
Application/ project documentation
Step F1
Identify related groups of data
List of Logical Files
Step F2
Classify logical groups of data
List of Internal Logical Files
List of External Logical Files
Step F3
Count Data and Record Elements
DET Count
RET Count
Data Complexity Formula
Weights Table
Internal
External
Step F4
Data Algorithm
List of points, by Logical File
Step F5
Add all data points
Data Points

Figure 2. Data measurement process.

tions for each of the GSC. Depending on the specifics of each GSC, this step produces one or multiple criteria for each GSC.

V2. The second step classifies each GSC into one of the six ordered classes (None to *Essential;* Table 3, righthand column) based on each GSC criteria classification rules described in IFPUG (1990).

V3. The third step, a degree of influence is assigned to each ordered class. This is currently a one-to-one relationship, and is exactly the same for each of the 14 characteristics.

V4. The fourth step is the addition of all the degrees of influence (N) for each of the 14 characteristics.

V5. The last step applies the adjustment algorithm with the empirical degree of influence formula ($0.65 + [0.01 * N]$) to obtain the value adjustment factor shown in Figure 1.

4. IDENTIFICATION OF FP MEASUREMENT SCALES

Overview of the Measurement Scales

This analysis focuses on two key aspects of any measurement process: identification and analysis of the use of the different scales within the various steps of the function point measurement model, with reference to the mathematical operations allowed for each type of measurement scale (Fenton, 1991; Zuse, 1991). The mathematical operations permitted for each scale type are summarized in Table 8.

1. Nominal scale: this scale is used to name objects or events for identification purposes only and has

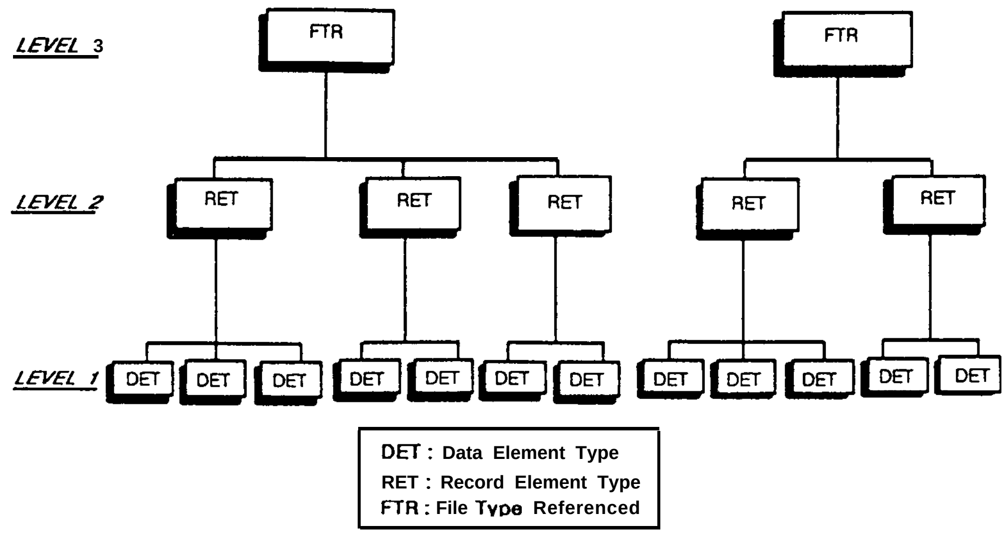

Figure 3. Data levels.

no quantitative implications associated with it. Only nonparametric statistics can be used.

2. Ordinal scale: this scale is used to order or rank items based on a criterion that can be subjective or, preferably, objective. Rank order statistics and all that apply to the nominal scale can be used.
3. Interval scale: this scale (also called the cardinal scale) is used to determine the difference between the ranks; it is continuous between two endpoints, neither of which is necessarily fixed. With this scale, the items can be distinguished and ranked, and the differences between ranks measured. Arithmetic mean and all statistics that apply to the ordinal scale can be applied.
4. Ratio scale: with this scale, nonnegative values can be used to multiply measurement values, and thus speak of item x having *n* times the value of item y with respect to a given attribute. A ramification of this, which is sometimes used to distinguish it from the interval scale, is the meaningful notion of a "zero" item with respect to the attribute. Percentage calculation and all statistics that apply to the interval scale can be applied.
5. Absolute scale: in addition to the properties of the ratio scale, the absolute scale has a unique origin from which to begin the measurement. Within this scale, entities can be counted (Fenton, 1991) and all of the above satistics apply. See Zuse (Chapter 8, 1991) for further discussion on this scale type.

Data Measurement Process

The uses of the measurements scales in function points are examined through an analysis of the data measurement process of Figure 2. As mentioned previously, the algorithm for the calculation of data points is usually described as a three-step calculation: logical files are first rated as being of low, average, or high functional complexity, then a number of points (weight) is assigned to each logical file depending on its rating of functional complexity, and then the points are added.

This descriptive procedure is, however, an oversimplification of the function point measurement process. To identify the types of scales and analyze their uses in this measurement process, the procedure must be broken down further, as follows:

1. In steps F1 and F3 of Figure 2, three different object types are counted: FTRs DETs, and RETs.

It should be noted that these three types of objects are not independent, but are organized in a hierarchical way (Figure 3): an RET is composed of one or many DETs, and an FRT is composed of one or many RETs. They represent three different levels of abstraction: the RET represents a structure of DETs and the FTR a structure of RETs. These three object types also have different semantics and properties. Steps F1 and F3 count these as three different object types, and the results of each addition can be used on an absolute scale.

2. Step F4 of Figure 2, referred to as the data algorithm, is examined in more detail, as follows.

2a. The first substep (Table 9) of this data algorithm consists of positioning the results of the previous additions of DETs and RETs into ranges specific to each object type. It must be noted that the results obtained in the mapping on these ranges cannot be added, but only ranked. Consequently, they do not qualify as a ratio or interval scale: this is merely a ranking process, and therefore it corre-

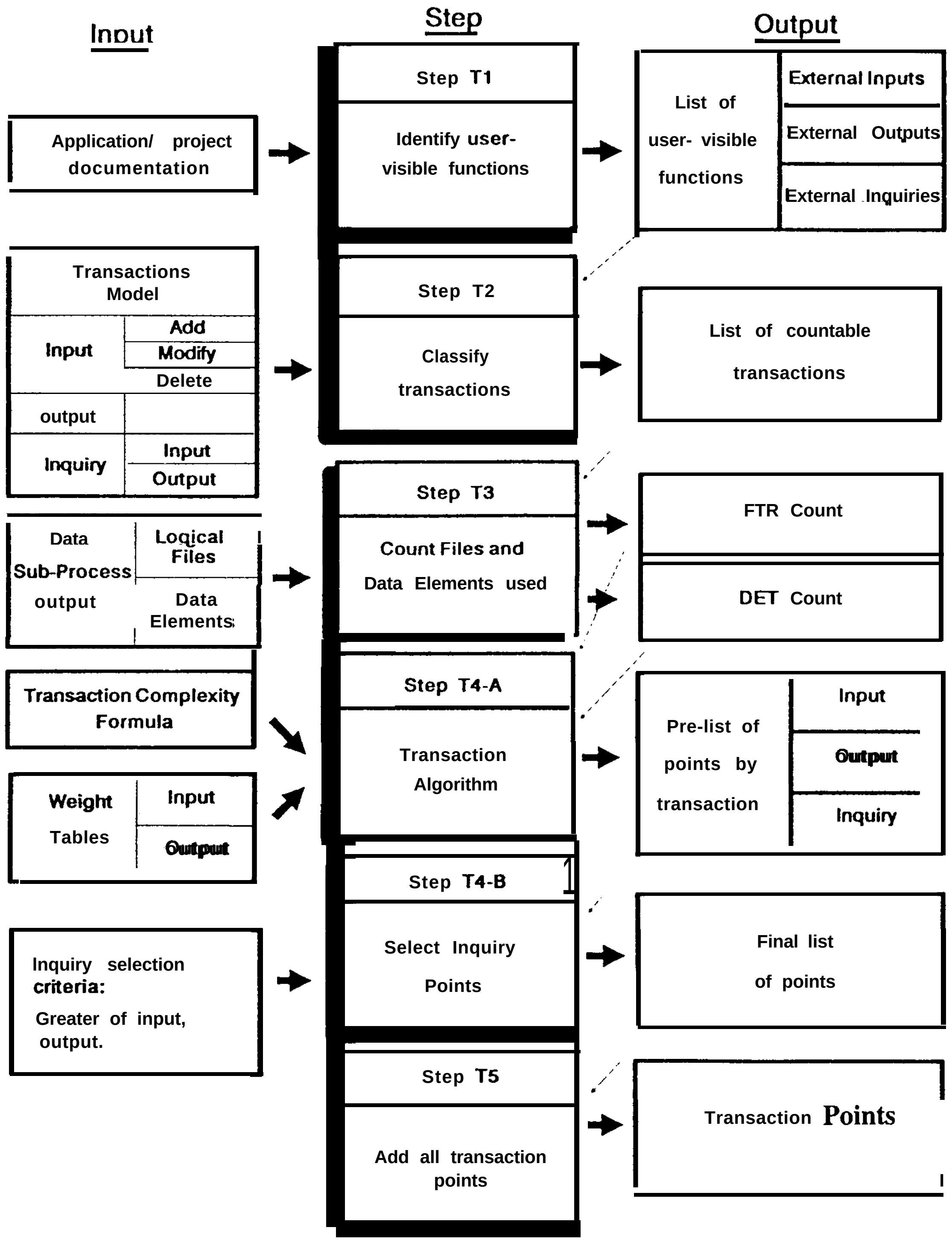

Figure 4. Transaction measurement process.

sponds only to an ordinal scale. From this point on, it can be said only that an object in Range 2 is bigger than an object in Range 1, and not that it is twice as big, or has twice as many elements. It must be observed that mathematical flexibility has been lost through this change of scale type. The distinct ordinal ranges of the two different object types, data elements and records, will be labeled with the following respective rank identifiers: Dl-D3 for DETs and Rl-R3 for RETs.

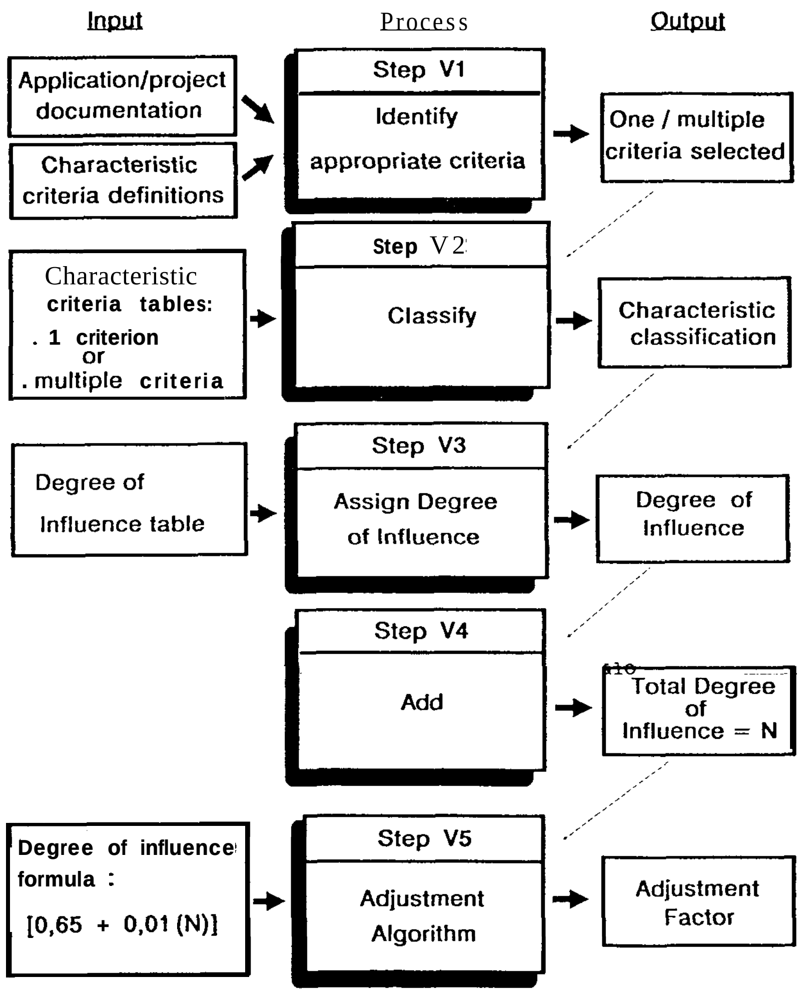
Input
Process
Output
Application/project documentation
Characteristic criteria definitions
Step V1
Identify appropriate criteria
One / multiple criteria selected
Characteristic criteria tables:
. 1 criterion or
. multiple criteria
Step V2
Classify
Characteristic classification
Degree of Influence table
Step V3
Assign Degree of Influence
Degree of Influence
Step V4
Add
Total Degree of Influence = N
Degree of influence formula :
[0,65 + 0,01 (N)]
Step V5
Adjustment Algorithm
Adjustment Factor

Figure 5. Adjustment measurement process.

2b. The next substep (Table 10) is much more complex from a measurement perspective: the ranks of two distinct types of objects (data elements and records) are taken as parameters of a function. This can be represented in mathematical notation as a function with two arguments, $\text{fn}(Di, Rj)$, where Di is the rank identifier for the data elements count and Rj the rank identifier for the record count. Although this is a normal process when building indices based on a variety of factors, it is not a standard measurements process. This substep does not produce results on an ordinal scale: although it could be argued that cell $(D1, R1)$ is smaller than cell $(D3, R3)$, it cannot be said that cell $(D1, R2)$ is either equal to or greater than cell $(D3, R1)$. The result of this substep indicates only the corresponding position on the matrix (Table 10), and it qualifies only on a nominal scale, with a loss of measurement information from the previous individual ordinal scales of the individual parameters of the function.

Table 8. Scales and Admissible Transformations

Scale Type	Admissible Transformations	Operations	Examples
Nominal	f unique	name, distinguish	colors, shapes
Ordinal	f strictly increasing monotonic function	rank, order	preference, hardness
Interval	$f(x) = ax + b$, $a > 0$	add	time of calendar, temperature in Celsius
Ratio	$f(x) = ax$, $a > 0$	add, multiply, divide	mass, distance, absolute temperature (degrees Kelvin)
Absolute	$f(x) = x$	add, multiply, divide	entities count

Table 9. Ranges of Data-Type Functions

Objects	Range 1	Range 2	Range 3
DET	D1:1–9	D2:20–50	D3: ≥ 51
RETs	R1:1	R2:2–5	R3: ≥ 6 +

2c. The next substep (Table 11) assigns a ranked qualifier (low, average, high) depending on their position on the two-dimensional matrix:

- Above the inverted diagonal: low
- On the inverted diagonal: average
- Under the inverted diagonal: high

This is referred to as the functional complexity assignment in function points. However, it does not explain what is understood by "complexity," and there has been no impact study conducted (besides expert judgements) to verify either the results of this classification or the classification process itself.

Furthermore, note that the diagonal does not correspond to usual mathematical notation. It does not express symmetry between the two axes, but merely a ranking of simultaneous increases on both axes (on an ordinal scale, with irregular and dissimilar ranges). Its purpose seems rather to express an equivalence of the rankings of the two axes when they are combined: for example, the smallest ranking on one axis combined with the highest ranking on the other axis is ranked similarly to the inverse ($[D1, R3]$ = high = $[D3, R1]$). For experienced programmers, this way of defining and quantifying the complexity of files seems to be intuitively valid; however, this is not a strict measurement process equivalence with respect to the mathematical operations allowed for each type of measurement scale!

Within function points, the result of this substep is treated as a rank within an ordinal scale (low, average, high). This implicit transformation from a nominal scale (substep 2b) to an ordinal scale (substep 2c) is not derived from the mathematical properties only: the end result of this substep would be admissible only in combination with some transformation based on proven theories substantiated by empirical research; this has not yet been the case with function points.

Table 10. Nominal Identifiers for Data-Type Functions

Data/Records	Rank D1	Rank D2	Rank D3
Rank R1	D1, R1	D2, R1	D3, R1
Rank R2	D1, R2	D2, R2	D3, R2
Rank R3	D1, R3	D2, R3	D3, R3

Table 11. Functional Complexity Matrix

Data/Records	Rank D1	Rank D2	Rank D3
Rank R1	Low	Low	Average
Rank R2	Low	Average	High
Rank R3	Average	High	High

2d. The next substep (Table 12) consists of assigning a weight to the qualified position on the matrix (low, average, high) based on whether it is an internal or an external logical file: the admissible values are 7, 10, and 1.5 for the internal files and 5, 7, and 10 for the external files. The results of this substep in function points are interpreted in the following way: an external file of high functional complexity will have a weight of 10 and is considered twice as large (!) as an external file of low functional complexity with its assigned weight of 5. In function points, the results of this substep are taken as ratio numbers that can be added together.

Even though this process again appears to be intuitively valid because of the weights assigned, it is not, by itself, a sound mathematical transformation. These weights represent, therefore, a transformation from an ordinal scale into a ratio scale. The values of these weights were initially derived based on expert judgement according to the "perceived different user values" (Albrecht, 1979) of the function types.

3. Finally, the last step, (F5 in Figure 2), the results of the assignments of weights to each data file are added together, taking for granted that the results of substep 2d are valid ratio numbers.

Because the ordinal scale results of substep 2c can not be used later on with the add operation in a measurement process, it can be seen that the set of weights of substep 2d was needed for adding the intermediate results of the measurement process.

From the empirical perspective, an interpretation of these weights is that there is an implicit hypothesis that they capture the relationship and establish a mapping to the amount of work effort required to deliver functionalities of the same type, but with different-sized attributes! What this does could be interpreted as follows: it transforms the measurement of the size of a function type, based initially on the number of data elements and records, into an

Table 12. Weights of the Data-Type Functions

File Types	Rank 1: Low	Rank 2: Average	Rank 3: High
Internal logical file	7	10	15
External logical file	5	7	10

assessment of the relationship with the work effort. This might qualify as a semantic transposition to arrive at an interval of the ratio scale number required for this step.

In summary, measurement information is lost from step 1 to substep 2a, and then again to substep 2b. Then, based strictly on the mathematical properties, nominal results of substep 2b could not have been added together in a measurement process unless combined through the successive transformations of substeps 2c and 2d.

Transaction and Adjustment Measurement Processes

All of the above comments on the uses of the scales apply to the transaction measurement process described in Figure 4 and to the adjustment measurement process described in Figure 5.

14. As a last step in function points, all the points are then added together, whether they come from internal logical files, external logical files, inputs, outputs, or inquiries. It must be observed that this has been made possible only by the assignment of weights to transform five different types of objects into one single object of a different type and an unspecified nature, that is, a function point. Again, this is done only through semantic transformations, and not through a strict measurement process with the proper use of the different types of scales.

The end results (unadjusted and adjusted function points) become, therefore, very difficult to interpret: there are so many dimensions involved and so many uses of different types of scales that the end result, which might look rather simple and reasonable, is, in fact, a potpourri of measurement scales. This supports Fenton's (1991) assertion that the end results may not be mathematically admissible, especially with respect to units and dimensions.

Therefore, at the present time, it would be more appropriate, based strictly on the admissible transformations of the measurement scales, to describe the end result of function points as an "index" rather than a "measure."

5. FUNCTION POINT INITIAL MAPPING SPACE

Measurement processes are an integral part of the scientific approach. To improve our understanding of these processes, descriptive models of their components must be constructed (Basili and Musa, 1991):

- The nature and characteristics of the processes and products
- The variations in these
- The strengths and weaknesses of each characteristic
- The mechanisms for predicting and controlling them

A measure is not a number per se, but the assignment of a number: it is a mapping between entities and their characteristics that are under observation. Whenever a characteristic is to be measured, it must be done with respect to a specific set of relationships (Fenton, 1991): it must be a mapping of a set of empirical relationships with a system of numerical relations. Measures must also be based on the mathematical discipline of the measurement theory. What is the appropriate mapping space to which the measure can be applied? Which models are used in the measurement process (Eijogu, 1991)?

In the previous section, it was illustrated that function points do not derive from a well-defined and proven theory; they are entirely empirically based on expert opinion. The above analysis of the measurement scales within function points has, through steps l-4 of the function points methodology, identified the existence of implicit transformations and implicit models without which the function point measurement process would be invalid.

It might be appropriate, therefore, to clarify the interpretation of a function point with respect to both measurement theory and its application within Albrecht's empirical model. The semantic analysis of both Albrecht's intention (implicit model) and the reference systems used to specify the structure and parameters of function points should provide a basis for a more appropriate interpretation of his definition from a measurement perspective.

The mapping, or the measurement space, of function point metrics clearly needs to be clarified if it is to be considered a valid measurement system. The domain of relationships being measured must be made more explicit if it is to be used properly, and possibly modified to expand its domain of applicability.

Albrecht's initial intention was to measure productivity, and to do so, he had to define and measure a product and a cost. The product analyzed by Albrecht was "function value delivered" and the objective was "to develop a relative measure of function value delivered to the user that was independent of the particular technology or approach used" (Albrecht, 1979). The result of Albrecht's measurement process met his objectives of productivity analysis. Various researchers have indeed verified that there is a strong empirical relationship

Table 13. Albrecht 79-Overall Software Engineering Environment

- A DP services organization of 450 people
- Application development for IBM's customers under contract
- Customers and service people located throughout the United States
- At any given time, 150-170 contracts under way
- Projects covered all industries
- Average contract size, 2-3 people
- Some projects required 15-20 people, a few, 35-40
- Each project carried out according to a formal development process
- Most projects carried out only a few specific tasks within the development process
- Design phases took ~ 20% of work hours, implementation, 80%
- All phases were measured, including design activities
- Project completion dates ranged from mid 1974 to early 1979
- Projects ranged in size from 500 to 105,000 work hours
- Of 1,500 contracts for that period, only 22 met the project measurement selection criteria

between between the size of an application measured with function points and work effort (Albrecht and Gaffney, 1983; Kemerer, 1987; Emrick, 1988a, 1988b; Desharnairs, 1988; Benyahia et al. 1990; Banker, 1989). However, his interpretation of a dimensionless number defined in function points should be revisited and reinterpreted from a measurement perspective.

The function point empirical model, including procedures and the set of weights, was derived from Albrecht's software engineering environment-the overall software engineering environment (Table 13) and the specific set of criteria to qualify the projects that were measured (Table 14)—as described in his 1979 article. An understanding of this reference context should help clarify the mapping domain of the initial empirical model from which the rules and procedures of the function point metrics were derived. It is important to realize that through this set of critera, Albrecht empirically defined a measurement space, or a model of the relationships under study, that described on only 22 projects of the 1,500 carried out during this reference period of mid 1974 to early 1979. It is also important to remember that this set of criteria defined a stable set of conditions, thereby explicitly attempting to limit the number of extraneous influences on the software processes and artifacts being analyzed.

Table 14. Albrecht 79-Project Measurement Selection Criteria

1.	Only complete projects that had proceeded through all phases from requirements definition to final systems test and demonstration and delivered a product to the customer were eligible.
2.	The whole project had to be carried out under IBM project management with consistent task definitions and management procedures.
3.	All work hours spent by IBM and customer's people had to be known and carefully accounted for.
4.	The functional factors had to be known.

6. CONCLUDING REMARKS

As reported earlier, there is a descriptive dissonance in saying that the size of an application can be expressed through a "dimensionless number." Function points interpretation should be reconsidered from a measurement perspective, and the issue of the expert judgments should be addressed as well as the measurement process itself with respect to the measurement scales and the transformations throughout the measurement steps. The following is a preliminary list of the implicit models embedded in function points through the expert judgments:

- The user's perspective
- The five-function-type model
- The high-level model of data function types (internal logical files and external interface files)
- The high-level model of transactions (input, output, inquiries)
- The lower level model of transactions (add, modify, and delete)
- The model of primary components (data, records, file structures)
- The model of the decision tables and their structure
- The model of the weights within the above-mentioned structures

Further research is required to explicitly describe these implicit models and investigate their suitability for and impact on the relationships under study.

Empirical research on function point metrics with respect to their relationship to work effort has focused mostly on the end product of the measurement process: the total count of either the unadjusted or adjusted function points. Further empirical research should include data for the intermediate steps of the function point methodology: if function points constitute a measurement system, and not only a recipe for constructing a dimensionless number, then each of the measurement steps, from the beginning to the end result, has a specific meaning and should contribute to the measurement process. Empirical research should be carried out to verify if each step in the function point measurement process adds information. If this is so, the transformations embedded in the expert judgments, or the implicit models, would transcend the mathematical operation and validate them through these transformations. It could even mean that some of the intermediate steps might be more meaningful if, in subsequent steps, information is lost instead of being added.

REFERENCES

Abran, A., and Robillard, P. N., Identification of the structural weakness of function point metrics, in *3rd Annual Oregon Workshop on Software Metrics,* Portland, Oregon, March 18, 1991.

Albrecht, A. J., Measuring application development productivity, in *Proceedings IBM Applications Development Symposium,* Monterey, California, October 14-17, 1979.

Albrecht, A. J., and Gaffney, J. E., Software Function, Source Lines of Code, and Development Effort Prediction: A Software Science Validation, *IEEE Trans. Software* Eng. SE-9, 639-648 (1983).

Banker, R. D., and Kemerer, C. F., Scale Economies in New Software Development, *IEEE Trans. Software Eng.* 15, 1199-1205 (1989).

Basili, V. R., and Musa, J. D., The Future Engineering of Software: A Management Perspective, *IEEE Comp., 90-96 (1991).*

Behrens, C. A., Measuring the Productivity of Computer Systems Development Activities with Function Points, *IEEE Trans. Software Eng.,* SE-9, 648-652 (1989).

Benyahia, H., Desharnais, J.-M., Hudon, G., and Charles, M., Adjustment model for function point scope factors -A statistical study, in *ZFPUG Spring Conference,* Lake Buena Vista, Florida, April 1990.

Conte, S. D., Dunsmsore, and Shen, V. Y., *Software Engineering Metrics and Models,* Menlo Park, CA: Benjamin/Cummings Publishing, 1986.

Cowderyoy, A., Jenkins, J., and Levy, W., State of the Art Survey. Volume 1: Effort and Size Estimation Models, MERMAID, ESPRIT Project P2046, 1989.

Desharnais, J.-M., Analyse Statistique de la Productivité des Projets de Développement en Informatique à Partir de la Technique des Points de Fonction, maîtrise informatique de gestion, Université du Quebec à Montreal, 1988.

Dunn, R. H., *Software Quality-Concepts and Plans,* Prentice-Hall, Englewood Cliffs, New Jersey, 1990.

Eijogu, L. O., Beyond Structured Programming: An Introduction to the Principles of Applied Software Metrics, *J. Struct. Progr. 11* (1990).

Eijogu, L. O., *Software Engineering with Formal Metrics,* QED Information Sicences, Wellesley, Massachusetts, 1991.

Emrick, R. D., Further analysis-Software development productivity-Second industry survey, in *ZFPUG Fall Conference,* Montreal, Quebec, Canada, September 1988a.

Emrick, R. D., Software development productivity-Second industry survey, in *ZFPUG Spring Conference,* Dallas, Texas, May 1988b.

Fenton, N. E., *Software Metrics-A Rigorous Approach,* Chapman & Hall, London, 1991.

Gaffney, J. E., The Impact on Software Development Costs of Using HOL's, *IEEE Trans. Software Eng. SE-12, 496-499 (1986).*

International Function Point Users Group, *Counting Practices Manual, Release 1.0,* IFPUG, Westerville, Ohio, 1986.

International Function Point Users Group, *Counting Practices Manual, Release 2.0,* IFPUG, Westerville, Ohio, 1988.

International Function Point Users Group, *Function Point Counting Practices Manual, Release 3.0,* IFPUG, Westerville, Ohio, 1990.

Johnson, J. R., 7'he *Software Factory-Managing Software Development and Maintenance,* 2nd edition, QED Information Sciences Inc., Wellesley, Massachusetts, 1991.

Jones, C., Building a Better Metric, *Computerworld Extra* (June 20, 1988a).

Jones, C., Feature points (function point logic for real time and system software), in *ZFPUG Fall 1988* Conference, *Montreal, Quebec, Canada, October* 1988b.

Jones, C., Measuring the Economic Productivity of Software, *Perspect. Technol. 2, 3-7* (1988c).

Kemerer, C. F., An Empirical Validation of Software Cost Estimation Models, *Commun. ACM 30, 416-429 (1987).*

Kemerer, C. F., Function point measurement reliability: A field experiment, in *ZFPUG Fall Conference,* San Antionio, Texas, October 1990.

Low, G. C., and Ross, J. D., Function Points in the Estimation and Evaluation of the Software Process, *IEEE Trans. Software Eng.* 16, 64-71 (1990).

Martin, J., *Rapid Application Development,* Macmillan Publishing, New York, 1991.

Pfleeger, S. L., An Investigation of Costs and Productivity for Object-Oriented Development, Ph.D. Thesis, George Mason University, Ann Arbor, Michigan, 1989.

Pressman, R. S., *Software Engineering, A Practitioner's Approach,* 2nd edition, McGraw-Hill, New York, 1987.

Pressman, R. S., *Making Software Engineering Happen, A Guide for Instituting the Technology,* Prentice-Hall, Englewood Cliffs, New Jersey, 1988.

Rudolph, E. E., Productivity in Computer Application Development, Department of Management Studies, University of Auckland, Auckland, New Zealand, 1983.

Rudolph, E. E., Precision of function point counts, in *ZFPUG Spring Conference, San Diego, California, April 1989.*

Schen, V. Y., Conte, S. D., and Dunsmore, H. E., Software Science Revisited: A Critical Analysis of the Theory and Its Empirical Support, *IEEE Trans. Software Eng. SE-9,* 155-165 (1983).

Symons, C. R., Function Points Analysis: Difficulties and Improvements, *IEEE Trans. Software Eng., SE-14, 2-11* (1988).

Wringley, C. D., A Model and Method for Measuring Information System Size, Ph.D. Thesis, University of British Columbia, Vancouver, British Columbia, Canada, 1988.

Zuse, H., *Software Complexity-Measures and Methods,* de Gruyter, Berlin, 1991.