Software Productivity Measurement Using Multiple Size Measures

Barbara Kitchenham and Emilia Mendes, *Member*, *IEEE Computer Society*

Abstract—Productivity measures based on a simple ratio of product size to project effort assume that size can be determined as a single measure. If there are many possible size measures in a data set and no obvious model for aggregating the measures into a single measure, we propose using the expression *AdjustedSize/Effort* to measure productivity. *AdjustedSize* is defined as the most appropriate regression-based effort estimation model, where all the size measures selected for inclusion in the estimation model have a regression parameter significantly different from zero ($p < 0.05$). This productivity measurement method ensures that each project has an expected productivity value of one. Values between zero and one indicate lower than expected productivity, values greater than one indicate higher than expected productivity. We discuss the assumptions underlying this productivity measurement method and present an example of its use for Web application projects. We also explain the relationship between effort prediction models and productivity models.

Index Terms—Software productivity measurement, software cost estimation.

1 Introduction

Productivity is the amount of output (what is produced) per unit of input used. In general, productivity is difficult to measure because outputs and inputs are typically quite diverse and are often themselves difficult to measure. In the context of software, productivity measurement is usually based on a simple ratio of product size to project effort (e.g., [16]). Thus, if we can measure the size of the software product and the effort required to develop the product, we have:

$$Productivity = Size/Effort. \quad (1)$$

Equation (1) assumes that size is the output of the software production process and effort is the input to the process. This can be contrasted with the viewpoint of software cost models where we use size as an independent variable (i.e., an input) to predict effort which is treated as an output. Equation (1) is simple to operationalize if we have a single dominant size measure, for example, product size measured in lines of code. MacCormak et al. used new lines of code per person day in a recent productivity study, noting that, "It is an imperfect measure of productivity but one that could be measured consistently" [16].

However, there are circumstances when there are several different effort-related size measures and there is no standard model for aggregating these measures. We recently collected data from Web companies where effort was strongly correlated with several different size measures (e.g., number of new Web pages, number of high effort functions, number of new images), each of which measured a different aspect of the overall size of a Web application and contributed significantly to a regression-based effort prediction model. When we have a number of significant size measures related to effort, it is difficult to determine how to construct a single size measure from the different individual measures. This means we cannot use (1) for productivity measurement.

In this paper, we present a method of constructing a productivity measure when effort is related to several different size measures. Section 2 summarizes a systematic literature review [14] used to identify previous work on software productivity measurement. Section 3 describes the motivation to our work, presents our proposed productivity measurement method based on multiple size measures, and discusses the assumptions underlying the productivity measure. We present the results of a productivity analysis using the new productivity measurement method in Section 4, followed by our conclusions in Section 5.

2 Related Work

We carried out a limited systematic literature review using the basic approach identified in [9], [15] in order to evaluate and interpret literature relevant to the following two questions:

1. What methods have been used to construct software engineering productivity measures when there is no single dominant size measure?
2. What explicit assumptions have software engineers made in productivity studies?

We used IEEExplore electronic database ensuring that our search was applied to journals, magazines, and

• *B. Kitchenham is with the National ICT Australia, Locked Bag 9013, Alexandria, NSW 1435, Australia, and with the Department of Computer Science, Keele University, Staffordshire, ST5 1BG, UK. E-mail: Barbara.Kitchenham@nicta.com.au, Barbara@cs.keele.ac.uk.*
• *E. Mendes is with the Computer Science Department, University of Auckland, Private Bag 92019, Auckland, New Zealand. E-mail: emilia@cs.auckland.ac.nz.*

Manuscript received 9 Mar. 2004; revised 17 Aug. 2004; accepted 10 Nov. 2004.
Recommended for acceptance by A. Mili.
For information on obtaining reprints of this article, please send e-mail to: tse@computer.org, and reference IEEECS Log Number TSE-0037-0304.

conference proceedings published over the last 10 years. Our review was more limited than a full systematic review because we did not follow up the references in papers, nor did we extend our search to include gray literature sources such as PhD theses and technical reports.

We experimented with several different search criteria and, in the end, used the following:

(commercial OR business OR industrial OR business-related) AND {G1}

(software project OR software development) AND {G2}

(method OR process OR system OR technique OR methodology) AND {G3}

(productivity OR production OR efficiency) AND {G4}

(measure OR metric OR attribute OR indicator OR variable OR measurement) AND {G5}

size {G6}

This search retrieved 104 papers, of which eight complied with our selection criteria ([2], [4], [7], [8], [18], [20], [21], [22]). Our search criteria missed three relevant papers ([26], [16], [23]). We believe two factors have contributed to the absence of these three papers from our search: the main factor is that IEEEXplore does not provide full-text search, using solely publication titles, keywords (index terms), and abstracts, making it difficult to achieve research completeness. The second factor is that publication abstracts are often incomplete, lacking relevant information. For example, [16] and [26] did not contain in its title/abstract/index terms any of the words from {G1} and {G6}; Reifer [23] did not contain in its title/abstract/index terms any of the words from {G1}, {G3}, or {G4}. Note that despite the ACM digital library providing full-text search, it lacks the flexibility for using a complex Boolean search expression, such as ours. Therefore, we decided not to use the ACM database.

Two papers were updates of previously reported research: [26] was an update of [22] and [7] was an update of [8]. In both cases, we considered only the most recent paper in our review, giving a total of nine papers to analyze. Data was extracted and results were tabulated. In this section, we only present a summary of the results obtained from our review due to shortage of space. However, the protocol we employed, the tabulated results, and the list of all journals, magazines, and conference proceedings used in our search are available in [14].

In spite of intending to identify research related to situations where there was no single dominant size measure, all papers, except for Stensrud and Myrtveit [26], used single size measures that either represented standard size measures for productivity assessment (e.g., function points, lines of code) or alternative size measures aggregated from other individual measures (e.g., System Meter [21], "Magnitude" [17], and Use Case Points [2]). Furthermore, all the papers, other than [26], that considered productivity measurement measured productivity as the standard size/effort ratio.

With the exception of Reifer [23], no researchers considered the problem of sizing Web applications. Reifer [23] proposed deriving an aggregated size measure (called Web Objects) from counts associated with Web application elements, including the number of new and reused Web pages, text files, animations, images, and functions. He proposed analyzing each element in terms of operator and operand counts and then combining the counts using Halstead's volume equation [6]. Reifer's approach has some similarity with ours and is contrasted with our approach in a later section.

Finally, Stensrud and Myrtveit [26] suggested using Data Envelopment Analysis (DEA) to investigate productivity differences (see [27] for a detailed explanation of DEA). The Data Envelopment method assesses productivity of projects relative to the most productive project that has similar output values (i.e., size measures). The idea of a relative measure for productivity is similar to our approach. However, the use of the most productive project as a baseline is based on the assumption used in economics that the most productive project is the one that used the best technology. This may not be the case in software where the most productive project may just be the easiest project. In addition, Kitchenham [9] has raised some objections to applying DEA to software projects. She pointed out that several assumptions underlying DEA were not always valid for software projects. In particular, DEA assumes that there is no measurement error, that all relevant input variables (i.e., measures) are included in the model, and that each data point represents an independent decision-making entity, with respect to allocating investment between labour and capital. None of these conditions are usual for software project data sets.

Apart from Reifer's Web Objects, none of the size measures presented in this section were developed explicitly to measure the size of Web applications. However, another possibility would be to use Principal Component Analysis (PCA) to reduce the dimensionality of the size data set to its principal orthogonal components [5]. Unfortunately, when the result of applying a PCA provides more than one orthogonal component, we still have the problem of having more than one size measure to use in our productivity measure.

The papers we reviewed made few references to their assumptions with respect to productivity measurement and modeling. Moser and Nierstrasz [21] made the point that reused components should not be included when measuring the size of an object-oriented application based on a framework. Morasca and Russo [20] make the point that, to construct a good productivity measure, size must be related to effort. Stensrud and Myrtveit [26] agree and, in addition, they note that Components-Off-The-Shelf (COTS) software projects have multidimensional outputs and, thus, require multidimensional size measures. We agree with all these points and believe they apply equally to Web applications.

3 Software Productivity Measurement

3.1 Motivation

One of the authors (Mendes) collected data on 54 Web projects from 25 Web application development companies. This data is part of the Tukutuku database[1] [19]. The aim of the data collection exercise was to investigate cross-company effort prediction and productivity models. The measures collected from the participating companies were based on the results of a survey of the Web sites of 133 Web application development companies that offered online quotes for Web projects [19]. This survey identified a number of different Web application factors that Web developers used to discuss Web products with their potential clients. For example, the most commonly used factors were the number of Web pages and a list of standard Web functions/features (e.g., shopping cart).

Based on the survey results, Mendes et al. [19] proposed a number of measures for cost estimation that appeared both to be available early in the life cycle and to be meaningful to clients of Web companies as well as the Web application developers. They validated the measures in discussion with the manager of a long-established (10 year-old) Web company. The measures included measures related to product size and a variety of process and staff characteristics (e.g., whether the project followed a documented process and the average years of experience of the development team). Size related measures were based on simple counts of the number of distinct elements in a Web application, i.e., counts of the number of new and reused Web pages, text files, animations, images, and functions. Thus, our selection of size measures is empirically based and we make no claim that we have identified all possible size measures for Web applications.

Reifer used similar concepts as the basis of his Web Object measure [23]. However, Reifer proposed analyzing each element in terms of operator and operand counts and then combining the counts using Halstead's volume equation [6]. Thus, in contrast to Web Objects, our size measures are much coarser measures, but can be obtained from clients early in the development process (i.e., during initial pricing).

When we used the Tukutuku database to investigate cross-company effort models [13], we experienced a number of issues that needed to be dealt with in order to obtain productivity measures:

- We had many different measures of Web application size, some of which were related to effort and some were not. Examples of size measures related to effort are the total number of Web pages and the total number of new images; examples of size measures not related to effort are the total number of video files and the total number of images reused from a library.
- Restricting our analysis to size measures, we found three measures jointly related to effort (i.e., all three of the measures were significant in a logarithmically transformed effort prediction model). The measures were the total number of Web pages, the total number of high effort functions, and total number of new images.
- There were other combinations of size measures that were almost as strongly correlated with effort as our "best-fitting" model. In particular, a model based on total number of new Web pages, total number of new high effort functions, and total number of new images appeared plausible as a model for Web applications exploiting some degree of reuse.
- The parameters in our size-based effort estimation model, reported in (6), showed strong evidence of nonlinearity, i.e., the multiplicative parameters for some size measures in the logarithmic regression model were significantly different from 1 (e.g., 0.442, 0.75).
- We obtained the total effort for each project in the data set. However, the data set included both projects that reused some elements (such as pages or functions) and projects that constructed all application elements from scratch. For reuse projects, we had no information about the proportion of total effort spent constructing new elements and the proportion of total effort spent on reused elements.

3.2 Productivity Measurement Method

Our suggestion for measuring productivity with multiple size measures is based on the idea that any *size-based effort estimation model* constructed by stepwise regression is **by definition** a function of effort-related size measures. Therefore, we can treat the *size-based effort estimation model* as an *AdjustedSize measure* and use the following equation to represent our proposed productivity equation:

$$Productivity = AdjustedSize/Effort. \qquad (2)$$

The $AdjustedSize$ measure includes only those size measures that *together* have a significant relationship with effort and it allows for the nonlinearity of the relationship, if necessary. This view of productivity has a number of obvious benefits:

- The expected value of productivity is one (since productivity is based on the ratio of estimated to actual effort).
- A value larger than one is an indication of better than average productivity, a value less than one is an indication of worse than average productivity.
- The regression analysis used to determine the effort estimation model can be used to construct upper and lower bounds on the productivity measure. This means we can assess whether the productivity achieved by a specific project is particularly good or bad. In addition, the width of the confidence interval gives us some idea of the precision of the productivity measure. Examples using upper and lower bounds are given in Section 4.2.

1. The Tukutuku data set is not in the public domain but can be made available to any researchers who would like to verify the results reported in this paper. The term Tukutuku is Maori for Web. The name was used because the data set arose from a New Zealand project investigating Web Application projects. The Tukutuku project's home page is located at http://www.cs.auckland.ac.nz/tukutuku.

Kitchenham et al. [12] suggested that the ratio of estimated effort to effort is a measure of estimate accuracy, which seems to contradict our interpretation of the ratio as a productivity measure. To a certain extent, the two interpretations simply reflect different viewpoints. A value less than one corresponds to underestimate which in turn can be interpreted as achieving a lower than expected productivity.

Furthermore, the goal of an estimation model is usually rather different from the goal of productivity measurement. Estimation models are intended to predict the effort of future projects. In our case, our productivity measure is intended to investigate the impact of various factors on the productivity achieved by projects in our current database, so our productivity measure is based on known size measures, not estimated size measures for new projects. In this context, "estimated effort" means the "expected effort," given the assumed regression model. It is not an effort prediction in the sense of a prediction for a future project.

The productivity measurement method presented in (2) can be applied to data sets in any application domain as long as there is a relationship between, at least some, size measures and effort, irrespective of whether the relationship is linear. However, the productivity measure obtained when using (8) would apply only to the projects in the current data base; it would not automatically apply to other projects from the same application domain unless the projects were a random sample from a well-defined population.

3.3 Productivity Measurement Assumptions

Our productivity measurement method assumes that an appropriate measure of size is strongly associated with effort. This is consistent with standard software productivity measures. If you have a size measure that is not directly related to effort, it cannot be used to construct a productivity measure. For example, suppose you have a product being maintained. It is not the total size of the product that is the main driver for maintenance effort, it is the amount of the product that is changed. Thus, a productivity measure based on total size divided by maintenance effort will not be valid because the output (i.e., total product size) will not be obtained as a result of the input effort (i.e., maintenance effort). The larger the software product and the smaller the amount changed, the worse the productivity measure will become. It will not be an accurate measure of maintenance productivity and will not be comparable for systems of different size and different maintenance loads.

The second assumption embedded in our method is much more controversial and is not consistent with current approaches to software productivity measurement. Our method assumes that possible economies and diseconomies of scale should be accounted for before undertaking productivity analysis. It arises because our productivity measure includes an adjustment for any economies or diseconomies of scale. If we have diseconomies of scale, as many software models suggest (e.g., COCOMO [3]), the relationship between effort and size is:

$$Effort = a \times Size^{b}, \tag{3}$$

where $b > 1.0$.

The mathematical form of this model means that large products take disproportionately more effort than small products. Therefore, if we construct a productivity measure based in (1) (i.e., $Size/Effort$), we will detect a productivity difference between large and small projects. This can be understood if we replace $Effort$ in (1) with $a \times Size^{b}$ to give:

$$Productivity = Size/a \times Size^{b} = Size^{(1-b)}/a. \tag{4}$$

If $b > 1$, we have diseconomies of scale and $Productivity$ decreases as $Size$ increases. If $b < 1$, we have economies of scale and $Productivity$ increases as $Size$ increases. If $b = 1$, we have a linear relationship and Productivity is constant, taking on the value $1/a$. If we use our approach, (2) would give:

$$Productivity = a \times Size^{b}/Effort. \tag{5}$$

Replacing $Effort$ by $a \times Size^{b}$ in (5) causes $Productivity$ to take on a constant value equal to 1. Thus, a productivity measure based on (5) removes the effect of diseconomies of scale by adjusting the size measure according to the exponential term b. Furthermore, in a conventional model, the term a is a productivity constant, whereas, in our approach, the term a is simply a constant of proportionality that converts our size function into the scale of the effort measure.

Our measurement approach can, therefore, only be used to investigate factors that affect the productivity of projects *after allowing for* diseconomies (or economies) of scale. We note that, by ignoring the impact of diseconomies (or economies) or scale, conventional software productivity analysis runs the risk of detecting factors that differ between large and small projects rather than factors that affect the productivity of all projects. For example, suppose we have data from many different countries and were interested in whether projects from country A were more productive than projects from country B but projects from country A were all small and projects from country B were all large. If there were diseconomies of scale, productivity analysis based on the conventional approach would find that country A was more productive than country B because country A submitted small projects. Our approach would find country A more productive than country B, if country A's small projects were on average more productive than the expected productivity of small projects in the database as a whole and country B's projects were on average less productive than the expected productivity of large projects in the database as a whole.

The third assumption underlying this approach is that it is easier to use a size-based regression model to construct a productivity measure than to attempt to construct a composite size model to measure size as a single aggregate measure. For example, the Albrecht function point measure [1] is the weighted sum of several different aspects of product size in terms of the flow of information across a system boundary and stored data (i.e., inputs, outputs, logical master files, inquires, and interfaces). The function point model provides a mechanism for aggregating the individual counts into a single size measure that can be used in (1). In the case of Web applications, the Web Object measure [23] has been constructed to play the same role as function points.

The choice of a single aggregate measure, compared with a size measure derived from a size-based effort estimation model, depends to a large extent on the validity of the aggregate measure. Without either an a priori model to support the aggregation or extensive empirical validation studies, it is difficult to be sure that the model is valid. For example, it is not clear that all the elements that contribute to a Web Object count are independently associated with effort. Furthermore, if size measures are correlated among themselves, aggregate size measures are vulnerable to double counting.

It is also important to recognize that our productivity measurement method is intended to allow productivity comparisons among projects belonging to the data set from which the effort estimation equation was obtained. We do not expect the particular productivity measure obtained for a specific data set to apply to other data sets (unless the data set was a random sample from a defined population). For each independent data set (whether obtained from a single organization or a group of organizations), the most appropriate estimation equation must be calculated before an appropriate productivity measure for that data set can be calculated.

There are several other assumptions related to our productivity measure, but all the assumptions apply equally to the standard productivity model shown in (1):

1. The numerator and denominator of the productivity measure are not close to zero. If both numerator and denominator are close to zero, ratio measures may approximate a Cauchy distribution.[2] A Cauchy distribution is an extremely unusual distribution for which the mean and variance are undefined, making standard productivity analysis problematic.
2. Staff effort is a good surrogate for project cost. This is not always true because effort does not allow for major differences in staff costs between different companies. For example, productivity comparisons used to assess the value of outsourcing to other countries are based on monetary costs, not staff effort.
3. Size measures based on the internal properties of different applications are comparable. This is equivalent to asking whether a line of code or a function point in one product is equivalent to a line of code or a function point in another product. There is no definitive answer to this question.
4. A productivity analysis is more useful than an extended analysis of effort (extending an effort regression analysis to include productivity factors as well as size factors). If we extend an effort regression analysis, we get the same results as we would get if we constructed a productivity measure using the size-based effort regression model and then performed a productivity analysis (this is explained in more detail in Section 4.2). The choice of a productivity analysis rather than an extended effort analysis is usually based on the view that it is easier to interpret a productivity analysis than a complex effort-based analysis. As previously mentioned, there is a slight difference between our productivity measure and the standard measure. If there are economies or diseconomies of scale, the standard measure would detect them during a productivity analysis, while our method would detect them in the effort equation.

2. http://en.wikipedia.org/wiki/Lorentzian_function.

4 Application of the Productivity Measurement Method

4.1 Productivity Measure Construction

We constructed the best fitting size-based effort prediction model, using manual forward stepwise regression [11], employing data on Web projects from the Tukutuku database. This is similar to the step-wise technique suggested by Kitchenham [11], but we used the additional variable plot facility in the STATA tool to assess the impact of each candidate variable on the residuals from the current model in each step. Since our size measures were highly skewed, they were transformed to a natural logarithmic scale to approximate a normal distribution. In addition, whenever a measure needed to be transformed but had zero values, the natural logarithmic transformation was applied after adding 1 to the measure (see (6)).

The best fitting model was (adjusted $R^2 = 0.6054$):

$$\begin{aligned} ln(AdjustedSize) = 2.262 + 0.442 \times ln(totWebPages) \\ + 0.753 \times ln(highefffns + 1) \\ + 0.130 \times ln(newimages + 1). \end{aligned} \tag{6}$$

This can be transformed into the following equation:

$$\begin{aligned} AdjustedSize = 9.6 \times (totWebPages)^{0.442} \\ \times (highefffns + 1)^{0.753} \\ \times (newimages + 1)^{0.13}. \end{aligned} \tag{7}$$

The variable definitions are shown in Table 1. In (6), the multiplicative value 9.6 can be regarded as the effort required to develop one Web page. Since 13 projects reused Web pages, (6) has an obvious limitation in that it does not distinguish new Web pages or new high-effort functions from reused Web pages or reused high-effort functions. However, using the Tukutuku database, reused Web pages were not included as a significant variable in the regression analysis.

Analysis of the best fitting model found five high-influence projects (using Cooks D criterion). These projects are listed in Table 2. Only one of the high-influence projects benefited from reuse (i.e., project 16). However, as a result of using (6), its effort estimate is considerably greater than

TABLE 1
Effort Model Variables

AdjustedSize	Total estimated effort to develop a Web application, measured in staff hours. Effort includes the effort spent in Requirements gathering, Design, Implementation (including adaptation and integration), Testing (integration and acceptance testing), Installation, any necessary re-work during development, Project management, any quality assurance carried during development, Configuration management, meetings, training and integration of new staff.
totWebPages	Total number of Web pages, which is the sum of total number of new Web pages, Web pages given by the customer and Web pages developed by third party. A Web page is a digital multimedia object as delivered to a client system (on a client/server architecture). It may incorporate applets or other elements active on either the client or server side. Examples of Web pages are html, htm, php, cfm, pdf and shtml. We exclude dynamically-generated pages.
highefffns	Total number high effort features/functions, which is the sum of total number of reused High effort features/functions without adaptation, total number of adapted High effort features/functions and total number of new High effort features/functions. A high effort function is a function that takes more than 12 hours to be developed, assuming a single developer.
newimages	Total number of new images developed for the project.

its actual effort. The effect of reuse is addressed from the viewpoint of productivity analysis in the following section.

Repeating the regression analysis excluding the high-influence projects does not result in any major changes to the model coefficients. Furthermore, the adjusted R^2 value for the regression model increases to 0.71 and the significance of the *newimages* variable is confirmed ($p < 0.042$). Thus, we conclude that the regression equation is reasonably stable for this data set and it is not necessary to omit the high-influence projects from any subsequent analysis.

Treating (7) as a weighted size function, we can construct a productivity measure:

$$\begin{aligned} Productivity = [\, & 9.6 \times (totWebPages)^{0.442} \\ & \times (highefffns + 1)^{0.753} \\ & \times (newimages + 1)^{0.130}]/ActualEffort, \end{aligned} \tag{8}$$

where *ActualEffort* is the total effort to produce the Web application, measured in person hours.

TABLE 2
High Influence Projects

Project Id	Total effort	Estimated Effort	Total Web pages	New Web pages
9	1786	283.42	100	100
16	363	2191.14	600	100
17	6	72.12	12	12
20	625	69.49	20	20
32	3150	357.53	22	22

4.2 Productivity Measurement

Using the Tukutuku data as an example, the productivity values constructed using (8) varied from a minimum of 0.11 to a maximum of 12.02. The mean value was 1.58, the standard deviation was 1.945, and the median was 0.98. The distribution of the productivity values is shown in Fig. 1. Constructing upper and lower bounds for the productivity values using the upper and lower bounds of the effort estimation model, we found three projects that had productivity values significantly different from 1:

1. Project 17 had a productivity value of 12.01 with 95 percent confidence limits (1.568, 92.141).
2. Project 20 had a productivity value of 0.111 with 95 percent confidence limits (0.014, 0.859).
3. Project 32 had a productivity value of 0.113 with 95 percent confidence limits (0.014,0.917).

All of these projects were identified as high-influence projects with respect to the best fitting effort estimation model (see Table 2).

4.2.1 Validation

After calculating our productivity values, the Company Director for one of the organizations that had contributed project data to Tukutuku (13 projects) was contacted in order to check whether the calculated productivity values provided accurate productivity measures for these 13 Web projects. He assessed productivity as a more complex attribute than we did, i.e., he understood productivity as a measure of how successful a project was, and success included factors such as requirements stability, customer satisfaction, and customer/staff personality type. When assessing the productivity for the 13 projects solely based on size and effort, his conclusion was that our productivity

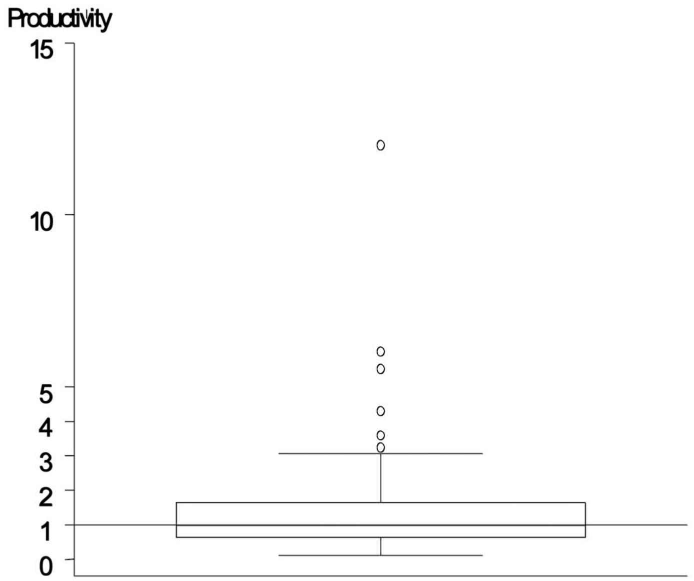

Fig. 1. Box plot of productivity values.

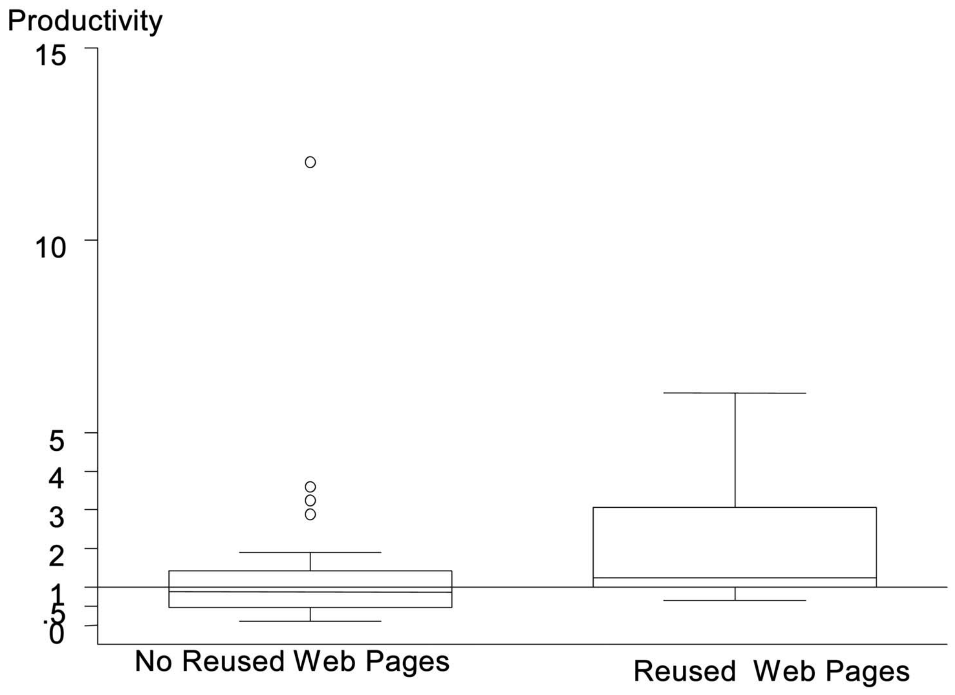

Fig. 2. Box plots of the productivity of projects that reused Web pages and those that did not.

measure provided reasonable productivity values for 12 out of the 13 projects (92 percent). The exception was a Web project (project A) considered by the Company Director as very productive and that showed low productivity using our measure. Further investigation of the data set revealed that this project was medium-sized (128 new Web pages, 2 high effort functions, and effort of 429 person hours) and developed by two people with an average experience of two years with the development languages employed. Project A had very similar characteristics to another project from the same Company (130 new Web pages, 3 high effort functions, and effort of 410 person hours) and was developed by three people with an average experience of five years. Our productivity measure suggested that both projects had low productivity; however, we can also understand the rationale used by the Company Director to grade Project A as a very productive project: Both projects used similar effort, one employed three very experienced staff and the other employed two less experienced staff, and most of the development languages used were the same.

4.2.2 Productivity Analysis

Once the appropriate productivity value has been calculated for each project, standard productivity analyses can be performed. An initial concern is the impact of reuse on productivity. We used a dummy variable to identify projects with reuse and investigated the productivity differences between the two groups of projects. The median productivity for the 13 projects that reused Web pages was 1.24, while the median productivity for projects that did not reuse Web pages was 0.87. Inspection of box plots of the productivity distribution for each group suggested some departure from the Normal distribution (see Fig. 2). The projects that did not reuse Web pages demonstrated a symmetric distribution but exhibited rather more outliers than would be expected of a Normal distribution. The distribution of the projects that did reuse pages was not symmetric about the median. We therefore used the nonparametric Kruskall-Wallis method to assess whether the difference between groups was significant. The chi-squared test statistic was 6.09 with 1 degree of freedom, which is significant at the 0.05 level (actual p value = 0.0.14). This confirms that reuse is having a significant effect on productivity and any further analysis of productivity should take account of this effect.

This might be thought to contradict our original effort estimation analysis where we found that the amount of reuse did not have a significant impact on effort. In fact, if we include reuse as a dummy variable in our effort estimation model, we find that the effect of the dummy variable is significant. Thus, in our data set, it is reuse *per se* not the amount of reuse that is important.

If we want to take account of the effect of reuse in subsequent analysis, we need to construct a productivity model. Given the nonnormal distribution of the productivity values, we should use a logarithmic transformation of the productivity values. However, if we transform (7), we obtain:

$$\begin{aligned} ln(Productivity) &= ln(AdjustedSize/ActualEffort) \\ &= ln(AdjustedSize) - ln(ActualEffort). \end{aligned} \quad (9)$$

Since $ln(AdjustedSize)$ is by definition $ln(EstimatedEffort)$, (9) implies that $ln(Productivity)$ is equal to the *negative of the residual of the effort estimation model*. This confirms the

TABLE 3
Model for Productivity Prediction (after Logarithmic Transformation)

Variable	Coef.	Std. Err.	t	P>|t|	[95% Conf. Interval]	
brazil	-0.1048	0.3162	-0.331	0.742	-0.7398	0.5303
reuse	0.6160	0.3020	2.040	0.047	0.00947	1.2226
constant	0.1487	0.1849	-0.804	0.425	-0.5201	0.22279

equivalence between productivity analysis and effort estimation modeling.

To continue the productivity analysis, we can construct a regression model including the dummy variable reuse and attempt to add other variables to the regression model. For example, companies submitting projects were interested in whether productivity was different among different countries. For categorical measures, such as the country of origin of the project, we can construct a dummy variable for each country and identify whether any country has an unusually high or low productivity. For example, if we construct a dummy variable to identify projects from Brazil and attempt to add that to the productivity model, including reuse, we obtain the model shown in Table 3. Since the coefficient for the dummy variable Brazil is not significantly different from zero, we conclude that projects from Brazil do not have an unusually high or usually low productivity values.

Alternatively, we can adjust our productivity values to account for reuse by performing a regression analysis using reuse as the independent variable and using the residuals from this analysis as our new productivity values [11]. Transforming back to the raw data scale, we can analyze the effect of different countries using the Kruskal-Wallis nonparametric analysis of variance. The box plots of productivity values for each country adjusted for reuse are shown in Fig. 3. Excluding Egypt and South Africa, which have only one project each, the Kruskal-Wallis analysis suggests that productivity is significantly different between countries (chi-squared = 11.69 with 5 degrees of freedom, p = 0.039). This effect is due to the three Canadian projects. If these projects are excluded, the analysis no longer detects any significant difference between countries (chi-squared = 6.96 with 4 degrees of freedom, p = 0.138). Furthermore, using a dummy variable to indicate Canadian

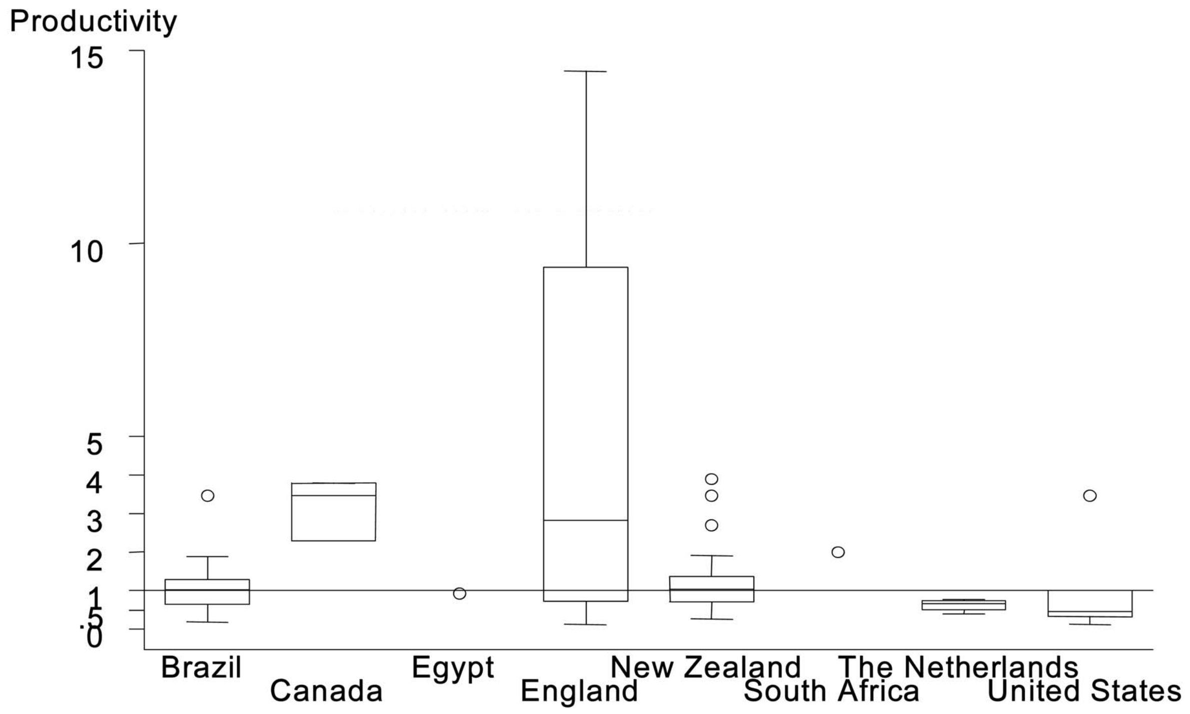

Fig. 3. Box plots of productivity (adjusted for reuse) for different countries.

TABLE 4
Relationship between $ActualEffort$ and $totWebPages$ (after Logarithmic Transformation)

Indep.Variable	Coefficient	Std. Err.	t	P>|t|	[95% Conf. Interval]	
ln(totWebPages)	0.6674508	0.1105	6.038	0.000	0.4455	0.8894
constant	2.366827	0.4532	5.223	0.000	1.4570	3.2767

projects, both reuse and Canadian origin are significant in a productivity regression model. However, our data set comprises data on projects volunteered by individual companies. It was not a random sample of projects from a defined population, thus, we cannot conclude that Canadian projects are in general more productive than projects from other countries. We can only confirm that the three Canadian projects in our data set were more productive than projects from other countries with respect to the variables included in our EstimatesEffort model—i.e., they tended to have less Webpages but more images than other projects. This issue is also a problem for conventional productivity analysis. If the data set used for productivity analysis is not a random sample, it is not clear whether any observed productivity differences apply to other projects. In particular, analysis is vulnerable to selection bias; for example, people may submit projects that achieved good productivity in order to look good or projects that achieved poor productivity in order to be able to easily exceed any benchmarks constructed from the submitted data.

4.2.3 Size Measures and Productivity Analysis

From (9), it is clear that if you use our approach to productivity measurement, you do not need to include size measures in your productivity analysis. The variables that jointly provide the best effort estimation model are already included in the $AdjustedSize$ measure and should not, therefore, be reused in the productivity analysis carried out after the productivity values have been obtained. Size measures not included in the best fitting estimation model will, by definition, not be factors that affect productivity. This is quite different from productivity analysis based on more conventional measures [17]. For example, suppose we had simply used the total number of Web pages to construct a productivity model:

$$SimpleProductivity = TotWebPages/ActualEffort. \quad (10)$$

For our data set, a regression analysis of effort against total number of Web pages (after transformation to the logarithmic scale) shown in Table 4 demonstrates an economy of scale (i.e., the coefficient of the variable $ln(totWebPages)$ is significantly less than 1). This implies that we will find a relationship between $ln(SimpleProductivity)$ and $ln(totWebPages)$. This is because, if

$$ln(ActualEffort) = a + b \times ln(totWebPages), \quad (11)$$

$$ln(SimpleProductivity) = ln(totWebPages) - ln(ActualEffort), \quad (12)$$

then

$$\begin{aligned} ln(SimpleProductivity) &= ln(totWebPages) \\ &- [a + b \times ln(totWebPages) \\ &= -a + (1 - b) \times ln(totWebPages). \end{aligned} \quad (13)$$

If b is significantly less than 1, then the term $(1 - b) \times ln(totWebPages)$ will be significantly different from 0, and the variable $ln(totWebPages)$ will appear as significant in the productivity model. To confirm this analysis, we show the regression analysis of $SimpleProductivity$ against $totWebPages$ (after the logarithmic transformation) in Table 5. A comparison of Tables 4 and 5 (where the coefficient terms have been deliberately kept to seven significant digits) confirms the relationship shown in (13), i.e.,

TABLE 5
Relationship between $SimpleProductivity$ and $totWebPages$ (after Logarithmic Transformation)

Indep. Variable	Coefficient	Std. Err.	t	P>|t|	[95% Conf. Interval]	
ln(totWebPages)	0.3325492	0.1105	3.008	0.004	0.1106	0.5545
cons	-2.366827	0.4532	-5.223	0.000	-3.2767	-1.4570

TABLE 6
Regression Model Using $ln(Productivity)$ as the Dependent Variable

Independent variable	Coef.	Std. Err.	t	P>|t|	[95% Conf. Interval]	
ln(ActualEffort)	-0.3945918	0.0684	-5.765	0.000	-0.53199	-0.2572
cons	1.943187	0.3524	5.514	0.000	1.2356	2.6507

$$ln(SimpleProductivity) = -(2.366827) + (1 - 0.6674508) \times ln(totWebPages), \quad (14)$$

$$ln(SimpleProductivity) = -2.366827 + 0.3325492 \times ln(totWebPages). \quad (15)$$

4.2.4 Effort and Productivity Analysis

It is important to realize that, unlike the size measures, you are quite likely to find a spurious relationship between productivity and *effort* as a result of the functional relationships between the productivity measure and the effort. This occurs because of the relationship shown in (9). If we attempt to find a relationship between productivity and effort on the logarithmic scale, we are looking for a relationship between $ln(ActualEffort)$ and

$$ln(Productivity) = ln(EstimatedEffort) - ln(ActualEffort),$$

and we know that a relationship exists between $ln(EstimatedEffort)$ and $ln(ActualEffort)$ because this is the relationship we found in the original effort-based regression analysis. In fact,

$$ln(Productivity) = ln(EstimatedEffort - ln(ActualEffort) = c + d \times ln(ActualEffort), \quad (16)$$

then

$$ln(EstimatedEffort) = c + (d + 1) \times ln(ActualEffort). \quad (17)$$

Equation (17) is obtained directly from (16). Equation (16) makes it clear that the better the relationship between $EstimatedEffort$ and $ActualEffort$ (i.e., the closer the value of $EstimatedEffort$ to $ActualEffort$), the more likely it is that $ln(Productivity)$ will not be significantly different from 0, which means that the additive constant c and multiplicative parameter d must also be 0. Thus, the *worse* the relationship between $EstimatedEffort$ and $ActualEffort$, the *more* likely it is you will find a significant relationship between effort and productivity. In our case, if we fit the model shown in (16), we obtain the regression coefficients shown in Table 6. If we fit the model shown in (17), we obtain the regression coefficients shown in Table 7. As expected, the constant coefficient term is the same for both models. The multiplicative term in Table 7 is obtained by adding one to the multiplicative coefficient in Table 6.

5 CONCLUSIONS

We present a software productivity measure that can be used when there are several size measures related to different aspects of a software product that are jointly significantly related to effort. The measure is easy to construct from a regression-based effort prediction model, and it is simple to interpret. However, unless the project data is a random sample from a defined population, the measure is applicable only to projects belonging to the

TABLE 7
Regression Model Using $ln(EstimatedEffort)$ as the Dependent Variable

Independent variable	Coefficient	Std. Err.	t	P>|t|	[95% Conf. Interval]	
ln(ActualEffort)	0.6054082	0.0684	8.846	0.000	0.46801	0.7428
Constant	1.943187	0.3524	5.514	0.000	1.2356	2.6507

data set on which it was constructed. This is, of course, true of any data-based effort estimation model.

The productivity measure has a built-in baseline. A value greater than one is indicative of good productivity, and a value less than one is indicative of poor productivity. Thus, this approach to productivity leads to a measure that is easier to interpret than more conventional productivity measures that do not have a built-in baseline.

Productivity analysis using our suggested measure follows the same steps as productivity analyses using simpler productivity measures. The main difference is that our approach to productivity measurement will not detect productivity differences between large and small projects because any economies or diseconomies of scale are accounted for in the measure itself. An important implication of our approach to productivity measurement is that it makes explicit the mathematical relationship between effort estimation and productivity modelling. Furthermore, our results indicate that conventional productivity analysis should recognise the impact of economies and diseconomies of scale on productivity analysis and avoid possible confounding between productivity differences and size differences.

This paper has discussed how to construct and manipulate productivity measures but has made the point that productivity analysis is restricted to analysis of a specific data set unless the data set is a random sample from a well-defined population. This is an outstanding problem for software data sets. We need to identify methods for drawing robust conclusions from nonrandom and quasirandom data sets (see, for example, [25]).

Acknowledgments

The authors would like to thank Ross Jeffery for his perceptive comments on a previous draft of this paper. Also, they thank all the companies that have contributed data to the Tukutuku project.

References

[1] A. Albrecht and J. Gaffney, “Software Function, Source Lines of Code and Development Effort Prediction: A Software Science Validation,” *IEEE Trans. Software Eng.*, vol. 9, no. 6, 1983.

[2] M. Arnold and P. Pedross, “Software Size Measurement and Productivity Rating In a Large-Scale Software Development Department,” *Proc. 20th Int'l Conf. Software Eng.*, pp. 490-493, Apr. 1998.

[3] B.W. Boehm, *Software Engineering Economics.* Prentice-Hall, 1981.

[4] S.R. Chidamber, D.P. Darcy, and C.F. Kemerer, “Managerial Use of Metrics for Object-Oriented Software: An Exploratory Analysis,” *IEEE Trans. Software Eng.*, vol. 24, no. 8, pp. 629-639, Aug. 1998.

[5] S. Elbaum, D. Gable, and G. Rothermel, “Understanding and Measuring the Sources of Variation in the Prioritization of Regression Test Suites,” *Proc. Seventh Int'l Software Metrics Symp.*, pp. 169-179, Apr. 2001.

[6] M.H. Halstead, *Elements of Software Science.* Elsevier, 1977.

[7] T.E. Hastings and A.S.M. Sajeev, “A Vector-Based Approach to Software Size Measurement and Effort Estimation,” *IEEE Trans. Software Eng.*, vol. 27, no. 4, pp. 337-350, Apr. 2001.

[8] T.E. Hastings and A.S.M. Sajeev, “A Vector-Based Software Size Measurement,” *Proc. Australian Software Eng. Conf. (ASWEC '97)*, pp. 7-15, 1997.

[9] B.A. Kitchenham, “Procedures for Performing Systematic Reviews,” Joint Technical Report Keele University TR/SE0104 and NICTA 0400011T.1, pp. 1-28, 2004.

[10] B.A. Kitchenham, “The Question of Scale Economies In Software—Why Cannot Researchers Agree?” *Information and Software Technology*, vol. 44, pp. 13-24, 2002.

[11] B.A. Kitchenham, “A Procedure for Analysing Unbalanced Datasets,” *IEEE Trans. Software Eng.*, vol. 24, no. 4, pp. 278-301, Apr. 1998.

[12] B.A. Kitchenham, L.M. Pickard, S.G. Macdonell, and M.J. Shepperd, “What Accuracy Statistics Really Measure,” *IEE Proc. -Software*, vol. 148, no. 3, pp. 81-85, 2001.

[13] B.A. Kitchenham and E. Mendes, “A Comparison of Cross-Company and Within-Company Effort Estimation Models for Web Applications,” *Proc. Empirical Assessment in Software Eng.*, pp. 47-56, May 2004.

[14] B.A. Kitchenham and E. Mendes, “Systematic Review of Software Productivity Measurement,” http://www.cs.auckland.ac.nz/emilia/srspp.pdf, 2004.

[15] B.A. Kitchenham, T. Dybå, and M. Jørgensen, “Evidence-Based Software Engineering,” *Proc. Int'l Conf. Software Eng.*, pp. 273-281, May 2004.

[16] A. MacCormack, C. Kemerer, M. Cusumano, and B. Crandall, “Trade-Offs between Productivity and Quality in Selecting Software Development Practices,” *IEEE Software*, pp. 78-79, Sept./Oct. 2003.

[17] K.D. Maxwell, *Applied Statistics for Software Managers.* Software Quality Institute Series, Prentice-Hall, 2002.

[18] K.D. Maxwell and P. Forselius, “Benchmarking Software Development Productivity,” *IEEE Software*, vol. 17, no. 1, pp. 80-88, Jan./Feb. 2000.

[19] E. Mendes, N. Mosley, and S. Counsell, “Investigating Early Web Size Measures for Web Cost Estimation,” *Proc. Int'l Conf. Empirical Assessment in Software Eng.*, pp. 1-22, 2003.

[20] S. Morasca and G. Russo, “An Empirical Study of Software Productivity,” *Proc. 25th Ann. Int'l Computer Software and Applications Conf.*, pp. 317-322, Oct. 2001.

[21] S. Moser and O. Nierstrasz, “The Effect of Object-Oriented Frameworks on Developer Productivity,” *Computer*, vol. 29, no. 9, pp. 45-51, Sept. 1996.

[22] I. Myrtveit and E. Stensrud, “Benchmarking COTS Projects Using Data Envelopment Analysis,” *Proc. Sixth Int'l Software Metrics Symp.*, pp. 269-278, Nov. 1999.

[23] D. Reifer, “Web-Development: Estimating Quick-Time-to-Market Software,” *IEEE Software*, vol. 17, no. 8, pp. 57-64, Nov./Dec. 2000.

[24] J. Rosenberg, “Some Misconceptions about Lines of Code,” *Proc. Fourth Int'l Software Metrics Symp.*, pp. 137-142, Nov. 1997.

[25] W.R. Shadish, T.D. Cook, and D.T. Campbell, *Experimental and Quasi-Experimental Designs for Generalized Causal Inference.* Houghton Mifflin Company, 2002.

[26] E. Stensrud and I. Myrtveit, “Identifying High Performance ERP Projects,” *IEEE Trans. Software Eng.*, vol. 29, no. 5, pp. 398-416, May 2003.

[27] S.C. Kumbhakar and C.A.K. Lovell, *Stochastic Frontier Analysis.* Cambridge Univ. Press, 2000.

Barbara Kitchenham is a professor of quantitative software engineering at Keele University and currently has a part-time position as a senior principal researcher with the National ICT Australia (NICTA). She has worked in software engineering for more than 20 years, both in industry and academia. Her main research interest is software metrics and its application to project management, quality control, risk management, and evaluation of software technologies. She is particularly interested in the limitations of technology and the practical problems associated with applying measurement technologies and experimental methods to software engineering. She is a Chartered Mathematician and fellow of the Institute of Mathematics and Its Applications. She is also a fellow of the Royal Statistical Society. She is a visiting professor at both the University of Bournemouth and the University of Ulster.

Emilia Mendes is a senior lecturer of computer science at the University of Auckland, New Zealand. She is the principal investigator in the Tukutuku Research project, which aims to gather data on Web projects developed worldwide and to use this data to develop cross-company/within-company Web cost estimation models and to benchmark productivity across and within Web companies. She is the director of the WETA (Web Engineering, Technology and Applications) research group. She has presented numerous lectures, conference presentations, and workshops on Web cost estimation and cochaired with Professor Kitchenham an industry workshop on Web cost estimation and productivity benchmarking last January (Auckland, New Zealand). She has active research interests in Web sizing and cost estimation, and also on Web quality and productivity measurement. She has collaborated with Web companies in New Zealand and overseas on Web cost estimation and usability measurement. She is a member of the IEEE Computer Society and the Australian Software Measurement Association.