http://ansinet.com/itj

ITJ

ISSN 1812-5638

INFORMATION TECHNOLOGY JOURNAL

ANSInet
Asian Network for Scientific Information
308 Lasani Town, Sargodha Road, Faisalabad - Pakistan

Information Technology Journal 7 (6): 830-839, 2008
ISSN 1812-5638

Exponential Effort Estimation Model Using Unadjusted Function Points

T.W. Koh, M.H. Selamat and A.A.A. Ghani
Faculty of Computer Science and Information Technology, University Putra Malaysia,
43400 UPM Serdang, Selangor Darul Ehsan, Malaysia

Abstract: In this study, we present a window-based exponential effort estimation model to predict the effort required in terms of man days by using Unadjusted Function Point (UFP) size measure and eliminate the usage of General Systems Characteristics (GSCs). A very comprehensive statistical analysis and test was carried out on large amount of quality project data in the International Software Benchmarking Standard Group (ISBSG) Release 9 dataset, which were collected by the International Function Points User Group (IFPUG) count approach in the process of model development. The effectiveness of the model was examined and reported in this study.

Key words: Functional size measure, effort estimation model, unadjusted function point, regression analysis

INTRODUCTION

In this competitive global market, it is widely accepted that software development organizations need to monitor and control the software process in order to develop high quality product and within the expected schedule. To monitor and control the process, we need to qualify and quantify software products, processes and resources (Fenton and Pfleeger, 1997).

Software product measures quantify properties, which are usually can be classified into internal or external attributes. Internal attributes of a product can be measured in terms of product itself such as size and complexity. On the contrary, external attributes usually denote properties that can be measured by taking into account how the product relates to its environment (Fenton, 1994). Examples of external attributes are maintainability and understandability.

Software size represents one of the most dominant internal attributes of a product. It has been employed in several effort or cost estimation models as a predictor of effort and cost needed to design and implement the software (Kemerer and Porter, 1992; Hastings and Sajeev, 2001; Anda *et al*., 2001). Thus, size measurement is one of the major tasks for planning software project development with effective cost and effort estimation. In general, there are three fundamental attributes suggested by Fenton and Pfleeger (1997) for describing software size: length, functionality and complexity. Length is the physical size of the product and functionality measures the functions supplied by the product to the end user. Complexity can be interpreted into four categories: problem complexity, algorithmic complexity, structural complexity and cognitive complexity, depending on this perspective.

Software industry argues that length is misleading and that the amount of functionality inherent in a product captures a better picture of product size. Especially for those who generate effort and cost estimation from requirement analysis stage often prefer to estimate functionality rather than physical size.

There have been several serious attempts to measure functionality of software products. The first and most used attempt in functional size measure was the one by Allan J. Albrecht, from IBM, in 1979. He proposed and developed his well-known methodology called Function Point Analysis (FPA) as a technology-independent measure of size. However, there are several problems with the function-points measure as described by Fenton and Pfleeger (1997).

There are huge amount of studies on software effort estimation models and techniques in which a discussion on the relationship between software functional size and effort or cost as a primary predictor. Similarly, in this study, we presents window-based exponential effort estimation model based on the unadjusted function point count and eliminated the use of technical complexity factor in order to solve the problems with the uncertainty inherent in the subjective sub-factor ratings which can have a significant effect on the final function point value. Our hypothesis is that without the use of technical complexity factors, window-based exponential effort estimation model is still able to provide accurate final effort estimation.

FUNCTIONAL SIZE MEASURE

Functional Size Measurement (FSM) affords its roots in the late 70s. First and most used attempt in the filed

Corresponding Author: Tieng Wei Koh, Faculty of Computer Science and Information Technology, University Putra Malaysia, 43400 UPM Serdang, Selangor Darul Ehsan, Malaysia

currently named FSM was the one designed initially by Allan Albrecht, an IBM researcher, in 1979. He has developed his well-known methodology called FPA and this method was aimed at overcoming some of the shortcomings of measures based on Lines of Code (LOC) for estimation purposes and productivity analysis, such as their availability only after implementation phase and their technology dependence. The FPA method was based on the idea of determining size based on functional requirements and from the end user's viewpoint, taking into account only those elements in the application layer that are logically visible to the user and not the technology used (Albrecht and Gaffney, 1983).

FPA was designed for business information system environment and has become a de facto standard in the Management Information System (MIS) community. However, it generated a large number of variants for both MIS and non-MIS environments, such as real-time, web and object-oriented. In the '90s, several extended FPA techniques have been developed; the four most popular FSM are included COSMIC FFP, Mark II FP, NESMA FSM and IFPUG FSM. The evolution of the FSM methods is shown in Fig. 1.

MARK II function point: The mark II method, introduced by Charles Symons in 1988, which aimed to improve on Albrecht's approach by better taking into account the internal complexity of data-rich business application software. This technique is the second most commonly used functional size measurement which is simple in concept, easy to apply, aligned with modern systems analysis methods, for intelligent software sizing and estimating as described in UKSMA (1998), besides function point analysis and has been utilized almost exclusively in the UK. Mark II uses the same basic parameter as FP in its calculations. Mark II, however, makes use of fewer parameters and was intended to:

- Reduce the subjectivity in dealing with files by measuring the number of entities and their performance as they move through the data structure.
- Modify the FP method to compute the same numeric totals regardless of application boundary as a single system or as a set of related sub-systems.
- Focus on the effort required to produce the functionality rather than on the value of the functionality delivered to the users.
- Add six additional complexity factors to the 14 General Systems Characteristics (GSCs).

NESMA functional size measure: The Netherlands Function Point Users Group (NEFPUG) was founded in 1989 and is the largest FPA user group in Europe. The NESMA maintains it own counting practice manual, which is compliant with and is valuable complement to IFPUG counting practice manual, which is available in (NESMA, 2001).

In 2004, NESMA published its latest counting handbook, version 2.2, while IFPUG published its counting practice manual, release 4.2. NESMA and IFPUG both use the same terminology, albeit in a different language. Both NESMA and IFPUG differentiate the same five types of user functions: ILGV (ILF), KGV (EIF), IF (EI), UF (EO) and OF (EQ). The rules for determining the type and complexity of a function are the same, with a few exceptions:

- External inquiry and external output
- Complexity of an external inquiry
- Implicit inquiry
- Code data
- Physical media
- Queries with multiple selections

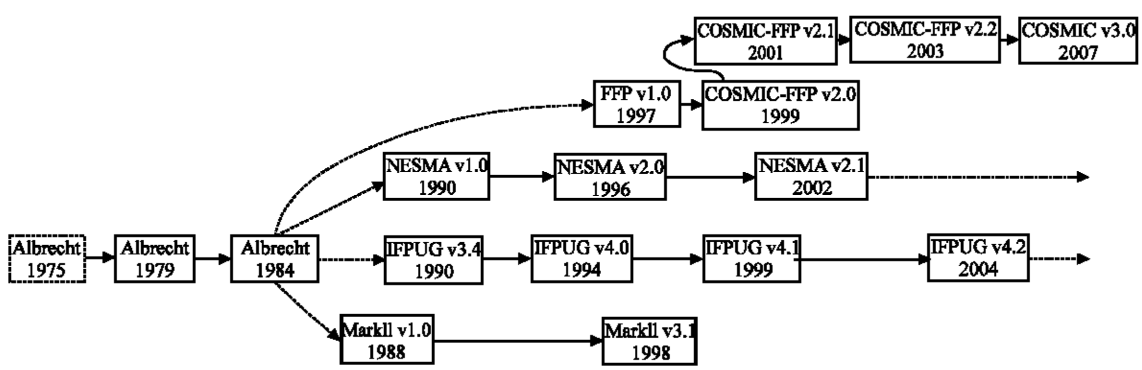

Fig. 1: Evolution of FSM methods

COSMIC full function point: The COSMIC is a group established by six countries: Australia, Canada, Finland, Netherlands, UK and the USA under supervision of Alain Abran and Chales Symons, with the aim to achieve an international standard set of software measurement. The COSMIC method (ISO 19761) is a functional size measurement method which generalizes the measurement process to address a variety of software domains especially MIS, real-time systems and infrastructure software such as operating system software through refinement of Full Function Point (FFP), MARK II and the FPA techniques. It was published in late 1999 and became stable with the publication of an International Standard definition in 2003. However, the method explicitly does not claim to measure the size of functionality, which includes complex data manipulation (i.e., algorithms) and does not attempt to take into account the effect on size of technical or quality requirements (Gao and Lo, 1996). The COSMIC function point data movement in contrast is tightly defined and the difficulties of ambiguous interpretation were not experienced in the field trials.

IFPUG functional size measure: Albrecht's original FPA method has evolved over the last 20 years into a method now known as IFPUG 4.2, through the original basic concepts and weighting methods have not changed since 1984. Over the same period, other methods have been put forward, each attempting to overcome weaknesses perceived in Albrecht's original model, or to extend its field or application (Gao and Lo, 1996). One of the most essential problems is the subjectivity measure on the complexity evaluation of the software in FPA as well described by Gao and Lo (1996) and Tichenor (2002). Hence, there are still many opportunities to continuously improve the methodology.

On the other hand, in early 2001, the International Organization for Standards (ISO) announced that recognition of FP as an international standard had taken a major step forward. By large majority, the national bodies comprising ISO approved the application for recognition filed by the IFPUG. Following resolution of the comments accompanying the votes of approval, function point will become the first software function sizing methodology to be recognized as an international standard (IFPUG, 2002).

Limitation of function point analysis: Allan Albrecht proposes function points as a technology-independent measure of size but there are several problems with this measure and users of the technique should be aware of its limitations, such as subjectivity in the complexity factor, double counting, counter-intuitive values, accuracy, changing requirements, technology dependence and application domain (Fenton and Pfleeger, 1997). The FP weights are justified by Albrecht as reflecting the relative value of the function to the user and determined by debate and trial. It is doubtful whether the weights will be appropriate for all users in all circumstances. FP weight limitations and its effect on software cost estimation were reported by Al-Hagri *et al.* (2003, 2004, 2005).

In addition, Jeffery *et al.* (1993) have shown that the UFP seems to be no worse a predictor of resources than the Adjusted Function Points (AFP) count. The used of 14 GSCs in FPA does not appear to affect the accuracy of the derived effort equations. So the 14 GSCs do not seem useful in increasing the accuracy of prediction (Symons, 1988). Among the study's conclusions was the notion that many feel that the 14 GSCs may not reflect the current software technology, as the 14 GSCs remained virtually unchanged at least since 1991 while technology has markedly changed. Recognising this, many function point users restrict themselves to the UFC. At least one major company does not directly include them as inputs into its popular commercial software estimation tool. Also, the current ISO consideration of function points does not include them (Tichenor, 2002).

Lokan (2000) reported that criticisms of GSCs and VAF are both theoretical and practical. Theoretical criticisms are that the construction of the VAF involves operations that are in admissible according to measurement theory; since complexity appears in computing unadjusted function points and again in the GSCs. Practical criticisms are that not all of the right things are counted as GSCs; when computing the VAF it is not appropriate to give all of the 14 GSCs the same weight; the VAF does not provide enough variation.

The selection of each factor of VAF is given by degree of influences using ordinal scale. This scale is determined from 0-5 as an ordinal scale which is limited to six values. Since the VAF ranged from 0.65 to 1.35, hence, the total UFP (TUFP) count can be changed by ±35%, as shown in Eq. 1:

$$\text{TUFP} = \sum_{i=1}^{5} \sum_{j=1}^{3} W_{ij}\, X_{ij} \qquad (1)$$

In this study, we present a window-based exponential effort estimation model to predict the effort required by using Unadjusted Function Point (UFP) size measure and eliminate the usage of GSCs in order to handle the uncertainty inherent in the subjective sub factor rating which can have a significant effect on the final FP-value.

EFFORT ESTIMATION MODEL DEVELOPMENT

The main objective for this research work is to reduce the uncertainty inherent in the subjective rating to GSCs in order to increase the confidence level and accuracy of final FP count. The assumption on this study is that Business Information Systems (BIS) suppose to have similar complexity level and hence, we taking the approach to eliminate the usage of 14 GSCs in final FP count.

The proposed model development process related statistical formulas stated in following section are based on statistical regression analysis rules and approach by Kutner *et al.* (2004) and Johnson and Kuby (2004).

Data sample: The project analyzed here come from the International Software Benchmarking Standards Group (ISBSG) Release 9 dataset. This is a public repository of data about completed software projects. The projects cover a wide range of applications, development techniques and tools, implementation languages and platforms. ISBSG believes that they are representative of better software development projects worldwide.

Although data sample is come form ISBSG, edit checks has been performed and plots prepared to identify gross data errors as well as extreme outliers. Difficulties with data error are especially prevalent in large data sets and need to be corrected or resolved before model building begins.

The repository contained data on 3024 projects, but only a sample of 450 projects remained base on following criteria:

- **Data quality:** Those data which was assesses as being sound with nothing being identified that might affect its integrity are selected
- **UFP rating:** The UFP count which was assessed as being sound with nothing being identified that might affect its integrity
- **Development type:** The new development projects were selected
- **Counting approach:** The IFPUG counting technique projects were selected

Establishing training samples and test samples: The sample, defined as a subset of population and the statistic can be a numerical value summarizing the sample data. Data sample were split into 2 sets with 70:30 ratio. These data sampling allocation is suggested by DMTeam (2006), they claims that a rule of thumb is to use 70% of the data for training and 30% for testing.

Training set defines a set of example used for learning that is to fit the parameters of the classifier and test set refers to as a set of examples used only to assess the parameter of a fully-specified classifier (Kutner *et al.*, 2004).

Three hundred fifteen projects were used as a training set to generate the proposed model using regression analysis and 135 projects will be used to validate the model.

Preliminary data analysis: In this study, we are using quantitative bivariate variables, it is customary to express the data mathematically as order pairs (x, y), where x is the input variable (sometimes called the independent variable) and y is the output variable (sometimes called the dependent variable). The data are said to be ordered because one value, x, is always written first. They called paired because for each x value, there is a corresponding y value from the same source. At here, function size, as an input variable x is measured or controlled in order to predict the total effort, output variable y.

Constructing a graph for quantitative data is required for the reason to display its distribution, where define as the pattern of variability displayed by the data of a variable. The distribution displays the frequency of each value of the variable. One of the simplest graphs used to display a distribution is the frequency histograms, a bar graph that represents a frequency distribution of a quantitative variable. The preliminary statistical studies of our two variables are shown in Table 1.

The frequency histograms with its normality curve, Fig. 2 and 3 present the raw training datasets of bivariate variables (function size and effort) used in this research work.

Summary in Table 1 and frequency histogram with its normality curve in Fig. 2 and 3, shown that the frequency distribution of the raw dataset for both variables (effort and function size) are not normal as shown in the normality curve for both variables are skew to the right. Hence, it is not suitable to use for model development using regression analysis.

Table 1: Preliminary statistical analysis

Statistical analysis		Effort	Function size
N	Valid	315.000	315.000
	Missing	0.000	0.000
Mean		6011.790	579.190
SD		7908.274	818.236
Variance		62540800.960	669510.328
Skewness		2.853	3.890
SE of skewness		0.137	0.137
Range		52008.000	7593.000
Minimum		164.000	6.000
Maximum		52172.000	7599.000

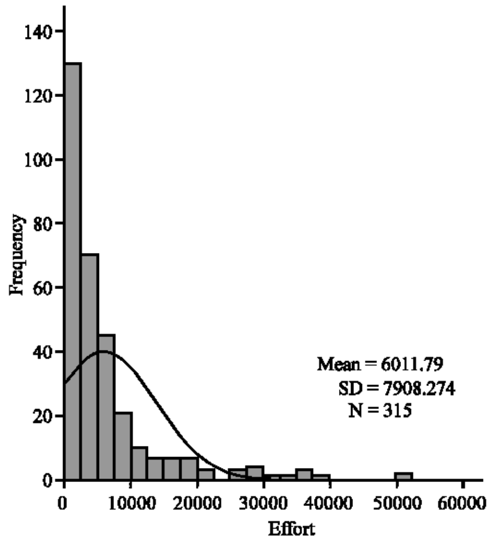

Fig. 2: Frequency distribution of 315 project's effort

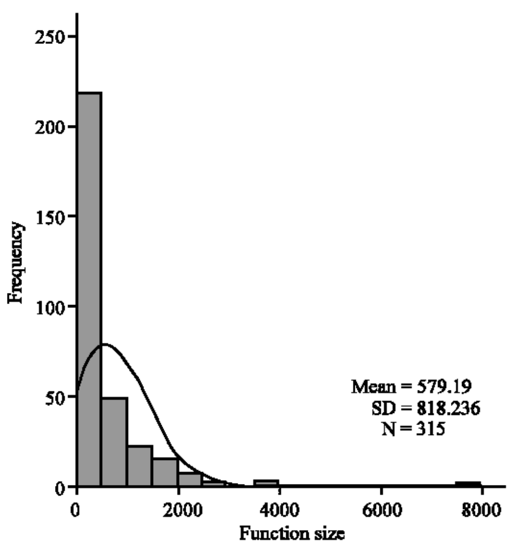

Fig. 3: Frequency distribution of 315 project's functional size

Linear correlation analysis: The primary purpose of linear correlation analysis is to measure the strength of a linear relationship between two variables.

Figure 4 shows relationships between input, or independent variables, functional size and output, or dependent variables, effort. It shown that there is no obvious correlation, or no positive nor negative relationship between the two variables.

The coefficient of linear correlation, r, is the numerical measure of the strength of the linear relationship between two variables. The coefficient reflects the consistency of

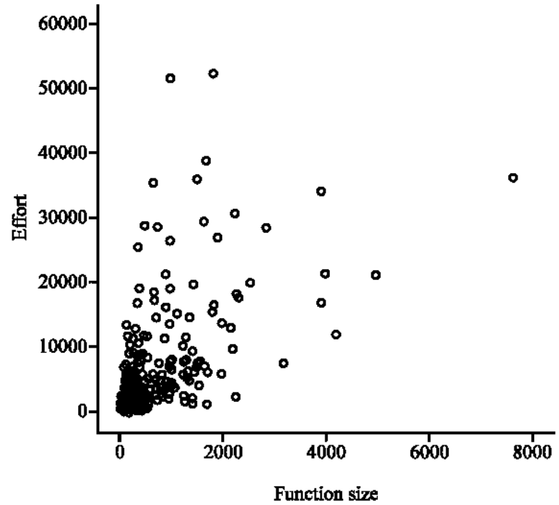

Fig. 4: Correlation between function size and effort

the effect that change in one variable has on the other. The linear correlation coefficient, r, always has a value between -1 and +1.

The value of r is defined by Pearson's product moment formula as in Eq. 2:

$$r = \frac{\Sigma(X-\bar{X})(Y-\bar{Y})}{(N-1)s_X s_Y} \qquad (2)$$

where, S_x and S_y are the standard deviations of the x and y variables.

Using the Eq. 2, the r of the raw data set, r = 0.721. The dataset has positive correlation between two variables and this is one of the factors for model generation based on regression analysis. Since data appear to violate an assumption (such as normality) and in order to have this kind of trend, we insist to carry out data transformation before move to next step.

Data transformation: It is mentioned earlier that a scatter plot of bivariate data shows curvature rather than a linear pattern, hence, a simple transformation of either the response variable y or the predictor variable x, or both are sufficient to make the simple linear regression model appropriate for the transformed data. The general pattern in a scatter plot is curved and monotonic as shown in Fig. 4, in this case, it is possible to find a power transformation for x or y or both, so that there is a linear pattern as well as achieve the normality for the transformed data. To straighten the plot, we use a transformation on x and y that is down the ladder, such as x and y = ln (x) and ln(y).

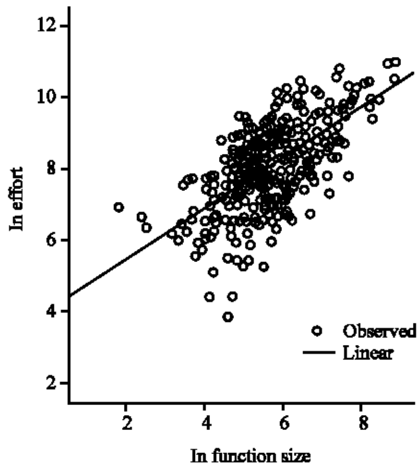

Fig. 5: Line of best fit on transformed data

Linear regression analysis: The equation of the line of best fit is determined by its slope (b_1) and its y-intercept (b_0) (Fig. 5). The values of the constants, slope and y-intercept, that satisfy the least squares criterion are found by using the formulas presented in Eq. 3 and 4:

$$b_1 = \frac{\Sigma(x-\bar{X})(y-\bar{Y})}{\Sigma(x-\bar{X})^2} \tag{3}$$

$$b_0 = \frac{\Sigma y - (b_1.\Sigma x)}{n} \tag{4}$$

Table 2 shows the values of two constants that will be used to create the prediction equation.

From the result generated by using statistical software- SPSS, the slope for the transformed data is 0.711 and the y-intercept is 4.005, as a result the linear equation shows in Eq. 5.

$$\ln(\check{y}) = 4.004 + 0.7110\ \ln(x) \tag{5}$$

The Eq. 5 can be simplified into Eq. 6 as below:

$$\begin{aligned} \text{In}(\check{y}) &= 4.004 + 0.7110\ln(\chi) \\ e^{\ln(\check{y})} &= e^{4.004+0.7110\ln(\chi)} \\ \check{y} &= e^{4.004} * e^{0.7110\ln(\chi)} \\ &= e^{4.004} * X^{0.7110} \end{aligned} \tag{6}$$

The linear model used to explain the behavior of linear bivariate data in the population represents in Eq. 7:

$$\check{y} = = \beta_0 + \beta_1 X + \in \tag{7}$$

Table 2: Constants values for the prediction equations

	Variables in the equation				
Variable	b	SE b	β	t	Sig t
ln Funct	0.711026	0.046626	0.652889	15.249	0.0000
(Constant)	4.004797	0.271584		14.746	0.0000
	ln Function	ln Function size			

Table 3: Variance of 313e's for transformed data

	df	Sum of squares	Mean square
Regre	1	206.79778	206.79778
Resid	313	278.34289	000.88927
F = 232.54664		Signif F = 0.0000	

β_0 is the y-intercept and β_1 is the slope. ϵ is the random experimental error in the observed value of y at a given value of x.

The regression line from the sample data gives us b_0, which is our estimate of β_0 and b_1, our estimate of β_1. The error β is a approximated as Eq. 8, the difference between the observed value of y and the predicted value of y, $\check{y}$, at a given value of x:

$$e = y\text{-}\check{y} \tag{8}$$

The random variable e (known as residual) is positive when the observed value of y is larger than the predicted value $\check{y}$; e is negative when y is less than $\check{y}$. The sums of the errors for all values of y for a given value of x is exactly zero (least square criteria). The variance of y about the regression line is calculated by using Eq. 9 and the result presented in Table 3, $s_e^2 = 0.89$ is the variance of 313e's for transformed data.

$$s_e^2 = \frac{(\Sigma y^2) - (b_0)(\Sigma y) - (b_1)(\Sigma xy)}{n-2} \tag{9}$$

Constructing a confidence interval for β_1: Here, we will discuss the procedure for constructing a confidence interval for β_1, the population slope of the line of best fit. The confidence interval is determined by Eq. 10.

$$b_1 \pm t(n-2, \alpha/2).S_{b_1} \tag{10}$$

Before we create the confidence interval for β_1, suppose the Eq. 11 should be discussed. This is the formula to calculate the variance of the error about the regression line:

$$S_{b_1}^2 = \frac{s_e^2}{\Sigma x^2 - \frac{(\Sigma X)^2}{n}} \tag{11}$$

In our bivariate data (x, y) data set, the variance among the b_1's is estimated as below:

$$S^2_{b_1} = \frac{0.8893}{10687.06527 - \frac{3237575.144}{315}} = 0.0022$$

The following steps proposed by Johnson and Kuby (2004) will be used to find the 95% confidence interval for population's slope, β_1.

- **The set-up**
  Describe the population parameter of interest.
  The slope, β_1, of the line of best fit for the population

- **The confidence interval criteria**
- Check the assumptions. The ordered pairs form a random sample and we will assume that the y values (Effort) at each x (function size) have a normal distribution
- Identify the probability distribution and the formula to be used. The student's t-distribution and Eq. 10
- State the level of confidence
  $1-\alpha = 0.95$

- **The sample evidence**
  Collect the sample information:
  $n = 315$, $b_1 = 0.7110$ and $s^2_{b_1} = 0.0022$

- **The confidence interval**
- Determine the confidence coefficients
  $t(df, \alpha/2) = t(313, 0.025) = 1.96$
- Find the maximum error of estimate
  We use equation (11) to find the maximum error
  $E = t\,(n\text{-}2, \alpha/2).S_{b_1}$
  $E = (1.96) * \sqrt{0.0022} = 0.0919$
- Find the lower and upper confidence limits
  b_1- E to b_1 + E
  0.7110 – 0.0919 to 0.7110 + 0.0919
  Thus, 0.6191 to 0.8029 is the 95% confidence interval for β_1

- **The result**
  State the confidence interval
  We can say that the slope of the line of best fit of the population from which the sample drawn is between 0.6191 and 0.8029 with 95% confidence.

Hypothesis testing: Now we are ready to test the hypothesis $\beta_1 = 0$. That is, we want to determine whether the equation of the line of best fit is of any real value in predicting y. For this hypothesis test, the null hypothesis is always H_0: $\beta_1 = 0$. It will be tested using the Student's t-distribution with df = n-2 degrees of freedom and the test statistic t* found using Eq. 12:

$$t* = \frac{b_1 - \beta_1}{S_{b_1}} \qquad (12)$$

One-tailed hypothesis test for the slope of the regression line: The following steps are proposed by Johnson and Kuby (2004).

- **The set-up**
- Describe the population parameter of interest
  β_1, the slope of the line of best fit for the population
- State the null hypothesis (H_0) and the alternative hypothesis (H_a)
  H_0: $\beta_1 = 0$ (this implies that x is of no use in predicting y; that is, y̌-y would be as effective)
  The alternative hypothesis can be either one-tailed or two-tailed. Since our slope is positive, as Fig. 5, a one tailed test is appropriate
  H_a: $\beta_1 > 0$ (we expect effort y to increase as the function size x increases)

- **The hypothesis test criteria**
- Check the assumptions
  The ordered pairs form a random sample and we will assume that the y values (effort) at each x (function size) have a normal distribution as in Fig. 6.
- Identify the probability distribution and the test statistics to be used
  The t-distribution with df = n-2 = 313 and the test statistic t* from Eq. 12
- Determine the level of significance:
  $\alpha = 0.05$

- **The sample evidence**
- Collect the sample information
  $n = 315$, $b_1 = 0.711$ and $s^2_{b_1} = 0.0022$
- Calculate the value of the test statistic
  Using Eq. 12, we find the observed value of t:

$$t* = \frac{0.711 - 0.0}{\sqrt{0.0022}} = 15.16$$

- **The probability distribution (Classical)**
- Determine the critical region and critical value.
  The critical region is the right-hand tail because H_a expresses concern for values related to positive. The critical value is found using critical values of student's t-distribution
- Determine whether or not the calculated test statistic is in the critical region
  t* is in the critical region, as shown in Fig. 7.

- **The result**
- State the decision about H_0:
  H_0 rejected

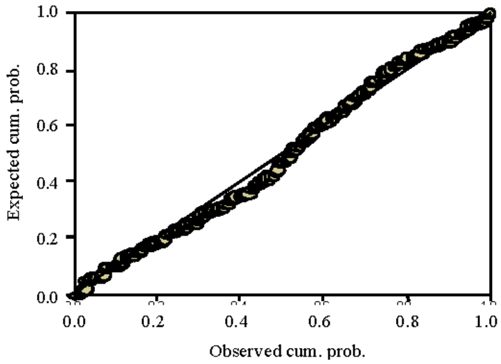

Fig. 6: Normal P-P Plot of regression standardized residual

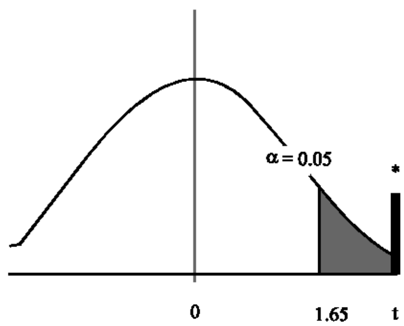

Fig. 7: One tailed critical region and critical value

- State the conclusion about H_a
  At the 0.05 level of significance, we conclude that the slope of the line of best fit in the population is greater than zero. The evidence indicates that there is a linear relationship and that the one-way function size (x) is useful in predicting the project effort (y)

MODEL PREDICTION POWER

The validity of the proposed model is our concern so far, the model is useful only if it is able to provide a reasonable accurate value in the early of the development life cycle.

The test set consists of 135 industrial real projects. These projects are business application with different sizes: small, medium and large, which are ranging from 3 to 19050 functional size.

The measurement approaches that are applied to the test set included:

- Error limits
- Mean magnitude of relative error (MMRE)
- Correlation coefficient
- Ratio of average error

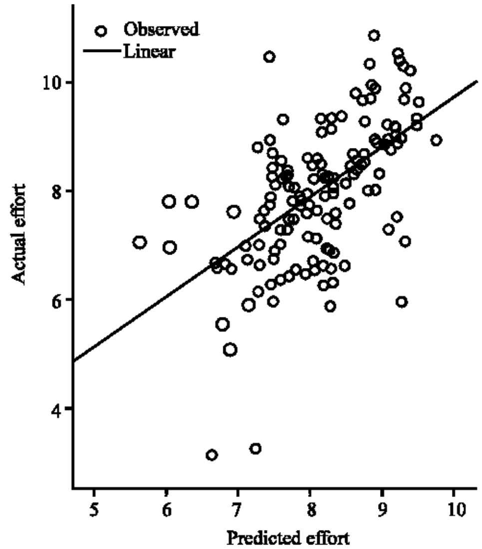

Fig. 8: Scatter plot between predicted and actual effort on proposed model

Table 4: Summary of measurement value

Measurement	Value
Ratio of average error	81%
MMRE	0.049
R^2	0.3296
Error limit	-3.02 to 3.92

The results show that the error between predicted value and actual value is relatively high (Table 4). We believed that this value will keep increasing as the number of test set goes up. Johnson and Kuby (2004) reported that the exact value of y is not predictable and we are usually satisfied if the predictions are reasonably close. This is also confirmed with the probability theory where the chance for hitting a single value under a normal curve is zero. In order to in crease the usage of the proposed model, this is where window-based estimation comes in.

With the confidence limits which was created using training set and apply it on test set, we notices that the proposed model give us 100% accuracy in predicting the final effort for business information projects, such that the actual effort values always fall within the predicted limits. This indicate that not only the data cleaning process which have been done is essential, but also the exponential equation and its confidence limits which was created using regression analysis is useful in the practical environment.

On the other hand, the correlation coefficient between predicted value and actual value seems not

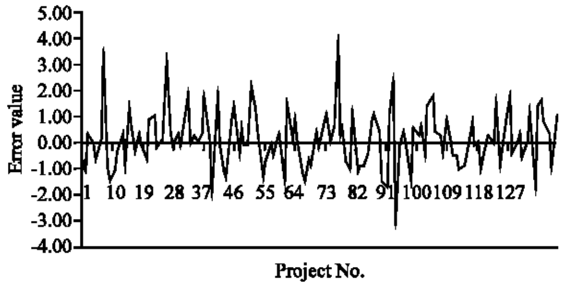

Fig. 9: Error curve for test set

convincing, but we believed that the value will goes up if the number of test set increase. The statistician, AHM Rahmatullah Imon claims that R^2 sometimes is misleading due to a single outlier in the data set. Further investigation on the test set should be carried out in near future.

Figure 8 shows the scatter plot between predicted and actual effort and the error curve for test set shown in Fig. 9.

CONCLUSIONS AND FUTURE DIRECTIONS

This research has concluded that the proposed model fulfilled the research objectives, eliminate the usage of 14 GSCs for FP count to support effort estimation at the early stage of the development life cycle by using the correction of 450 projects for effort estimation model development and testing. The proposed model can be use with confidence for all kind of business information systems development and it is very important for contract negotiations, especially for those organizations which are still relatively new in the industry.

The proposed effort estimation model shows its major advantage in reduce the subjectivity rating for 14 GSCs which will influence the accuracy of final FP count. With the use of upper limit and lower limit to the model has increase the confidence level for project planning and scheduling, especially for those who are new to FP counting approach. Besides, we also believe that the proposed model is much economy than the original FP count, which is combination between UFP and GScs.

There are many opportunity for researcher to conduct meaningful function point research. The two topics suggested below have potential of improving the accuracy of function point counts and reduce the variances we experience in our software business forecasting models based on function point measures of software size.

Expand the function point analysis of algorithms. An algorithm is a series of equations solved in logical sequence to produce an external output. Function point counters, software developers and others occasionally encounter algorithms embedded in software. Sizing these algorithms using function point analysis can result in more accurate measures of application size and improve quality in forecasting costs, schedule and quality. It can also improve the confidence of developers who are new to the function point methodology as they see all of their mathematical work is recognized and measured.

Test the proposed estimation model for other application domain. Our effort estimation model for function point measure were created based on majority business application, hence, the constant that created not necessary accurate for other application domain. This can be done by collecting other domain application dataset and test on the proposed model.

REFERENCES

Al-Hajri, M.A., A.A.A. Ghani, M.H. Selamat and M.N. Sulaiman, 2003. Multiple components for general system characteristics to adjust the function point calculations accuracy. Advance Technology Congress, May 20-21, Putrajaya, Malaysia.

Al-Hajri, M.A., A.A.A. Ghani, M.H. Selamat and M.N. Sulaiman, 2004. New complexity weights for function point analysis using artificial neural networks. Ph.D. Thesis, University Putra Malaysia.

Al-Hajri, M.A., A.A.A. Ghani, M.H. Selamat and M.N. Sulaiman, 2005. Modification of standard function point complexity weights system. J. Syst. Software, 74: 195-206.

Albrecht, A.J. and J.E. Gaffney, 1983. Software function, source lines of code and development effort prediction: A software science validation. IEEE Trans. Software Eng., SE-9: 639-648.

Anda, B., H. Dreiem, D.I.K. Sjoberg and M. Jorgensen, 2001. Estimating software development effort based on use cases-experiences from industry. Proceedings of the 4th International Conference on the Unified Modeling Language, Modeling Languages, Concepts and Tools. LNCS., Vol. 2185, October 01-05, Springer-Verlag London, UK., pp: 487-502.

COSMIC, 2006. The common software measurement international consortium. http://www.cosmicon.com/.

DMTeam, 2006. Sampling nested tables. Data Mining Community Site: http://www.sqlserverdatamining.com.

Fenton, N., 1994. Software measurement: A necessary scientific basis. IEEE Trans. Software Eng., 20: 199-206.

Fenton, N.E. and S.L. Pfleeger, 1997. Software Metrics: A Rigorous and Practical Approach. 2nd Edn., International Thomson Publishing, Boston, ISBN: 0-534-95425-1.

Gao, X. and B. Lo, 1996. A modified function point method for cal systems with respect to software cost estimation. IEEE Proceedings on the 1996 International Conference on Software Engineering: Education and Practice, Jan. 24-27, Dunedin, New Zealand, pp: 212-219.

Hastings, T.E. and A.S.M. Sajeev, 2001. A vector-based approach to software size measurement and effort estimation. IEEE Trans. Software Eng., 27: 337-350.

IFPUG, 2002. Function point analysis nears approval as first international standard (ISO) for software functional size. http://www.ifpug.org/about/ISOPress.htm.

Jeffery, D.R., C.G. Low and M. Barnes, 1993. A comparison of function point counting techniques. IEEE Trans. Software Eng., 19: 529-533.

Johnson, R. and P. Kuby, 2004. Elementary Statistics. 9th Edn., Thomson Brooks/Cole, Belmont, CA, ISBN: 0-534-39915-0.

Kemerer, C.F. and B.S. Porter, 1992. Improving the reliability of function point measurement: An empirical study. IEEE Trans. Software Eng., 18: 1011-1024.

Kutner, M.H., C.J. Nachtsheim and J. Neter, 2004. Applied Linear Regression Models. 4th Edn., McGraw Hill, New York, ISBN: 978-0256086010 .

Lokan, C.J., 2000. An empirical analysis of function point adjustment factors. Inform. Software Technol., 42: 649-659.

NESMA, 2001. Function point analysis for software enhancement. Professional guide of the Netherlands Software Metrics Users Association, Version 1.0. http://www.nesma.org.

Tichenor, C., 2002. Recommendations for further function point research. Longstreet Consulting Inc. http://www.SoftwareMetrics.Com/Articles/Tichenor.htm.

UKSMA, 1998. Mk II function point analysis counting practices manual Version 1.3.1. UKSMA Metrics Practices Committee, UK.