Full citation: MacDonell, S.G., Shepperd, M.J., & Sallis, P.J. (1997) Metrics for database systems: an empirical study, in Proceedings of the Fourth International Symposium on Software Metrics (Metrics'97). Albuquerque NM, USA, IEEE Computer Society Press, pp.99-107.
doi: 10.1109/METRIC.1997.637170

Metrics for Database Systems: An Empirical Study

Stephen G. MacDonell
Computer and Information Sciences
University of Otago
P.O. Box 56
Dunedin, New Zealand
+64 3 479 8142
stevemac@commerce.otago.ac.nz

Martin J. Shepperd
Department of Computing
Bournemouth University
Talbot Campus
Poole, BH12 5BB UK
+44 1202 595034
mshepper@bmth.ac.uk

Philip J. Sallis
Computer and Information Sciences
University of Otago
P.O. Box 56
Dunedin, New Zealand
+64 3 479 8142
fillup@otago.ac.nz

Abstract

An important task for any software project manager is to be able to predict and control project size and development effort. Unfortunately, there is comparatively little work, other than function points, that tackles the problem of building prediction systems for software that is dominated by data considerations, in particular systems developed using 4GLs. We describe an empirical investigation of 70 such systems. Various easily obtainable counts were extracted from data models (e.g. number of entities) and from specifications (e.g. number of screens). Using simple regression analysis, a prediction system of implementation size with accuracy of MMRE = 21% was constructed. This approach offers several advantages. First, there tend to be fewer counting problems than with function points since the metrics we used were based upon simple counts. Second, the prediction systems were calibrated to specific local environments rather than being based upon industry weights. We believe this enhanced their accuracy. Our work shows that it is possible to develop simple and useful local prediction systems based upon metrics easily derived from functional specifications and data models, without recourse to overly complex metrics or analysis techniques. We conclude that this type of use of metrics can provide valuable support for the management and control of 4GL and database projects.

Keywords: Metrics, entity-relationship models, 4GL, empirical analysis, prediction systems.

1. CONTEXT AND RELATED WORK

Software metrics — that is, measures derived from aspects of software engineering products or processes — have been promoted as useful adjuncts for the software developer since the early 1970s. Whilst the results have been mixed there is no denying the considerable amount of activity undertaken beneath this umbrella. Generally, the emphasis has tended to be upon procedural aspects of systems. DeMarco [5] made a distinction between systems that were characterised as being ‘function strong’ and those that were ‘data strong’. An example of the former would be the control software for a robot arm whilst a management information system would be an example of the latter. In practice systems contain elements of both functionality and data, however, in many cases either the functionality or the data tends to dominate. Most metrics research has concentrated been upon ‘function strong’ systems or the function aspects of systems. This is particularly evident for the design stage where measurement of data-oriented models has been neglected. Another specific area of neglect is fourth-generation languages (4GLs) which tend to be utilised for ‘data strong’ applications.

Why measure data aspects of a system? The size and nature of the data model and its associated transaction requirements — implicit or explicit — contribute to many aspects of a system including its size and the amount of effort to develop. Measuring aspects of the data model and its manipulation within data-centred transactions can potentially help control and predict these aspects of the software development process. In particular, measurements derived from a data model might be used as inputs for prediction systems that can be used to make the process of estimating size and effort to develop a software system more reliable at a relatively early stage in a project.

The earliest published attempt to measure a data model was by DeMarco [5] who proposed a so-called bang metric for data strong systems. This is based upon the entity-relationship (ER) diagram. Unfortunately, the authors are unaware of any published independent validations of prediction systems based upon the Bang metric. At about the same time Albrecht published his work based upon function points [1]. This is based upon the amount of data which crosses a system boundary plus internal file accesses. A subsequent adaptation, known as Mk II function points [17], is more closely based on transactions applied to ER models. In both cases there have been extensive attempts to use these measures to predict system size and development effort. However, the results of using function points are quite mixed. Jeffery *et al.* [9] found that a prediction system based upon function points was able to ‘explain’ less than 40% of the variation in effort. Kemerer [10] reported slightly better results in that function points were able to account for 55% of the variation in his independent validation. Although significant effort has been devoted to strengthening the counting practices associated with function points [7], questions of subjectivity and measure interdependence remain. Moreover, function point counting is quite a complex process that requires a degree of training. This in itself provides some motivation for the consideration of new approaches.

More recently, Verner and Tate [18] reported upon their attempts to predict the size and effort required for a 4GL implementation. Their solution was to use function points to estimate the size of the application, convert this into lines of code (LOC) and then to use this figure as an input to the COCOMO model [2]. The results, especially the size prediction, appear to have been quite accurate although a note of caution is required since Verner and Tate were only studying a single system.

Bourque and Côté [3] describe an empirical study where they attempted to predict the size of 4GL systems based upon various metrics derived from an ER diagram. Using linear regression they were able to develop effective prediction systems although they noted the need to calibrate the models to the specific measurement environment. A similar approach was suggested by Ince *et al.* [8] and Gray *et al.* [6] and indeed our data collection includes the raw counts required for the more complex synthetic metrics proposed by these authors. However, the prediction systems implied by their work remain unvalidated.

The aim of our investigation is to explore the possibility of developing useful prediction systems for software size for applications dominated by data considerations, in this case 4GL systems. In particular, we are looking for simple metrics and an easy method of developing prediction systems without recourse to complex analysis or the need for specialist input. If achieved, these aims should help to deliver an approach that is generally applicable to data-strong systems, whilst also providing results that are straightforward to interpret. The focus of our work is empirical; we believe it important to validate, as well as propose, models. The remainder of this paper goes on to describe the background to our study, the data collection, the analysis and concludes with discussion of our findings, how they relate to other work and how the results might be utilised by practitioners.

2. DETAILS OF THE STUDY

2.1. General system characteristics

The systems comprising the sample were built over a period of five years by groups of senior students in the Department of Information Science at the University of Otago. Every system was built to satisfy the real requirements of an external client, normally a small business or a department in a larger organisation. The students were required to use a prototyping process in development, meeting with their client on around three occasions over the nine week development period. Each system addressed transaction processing, data retrieval and reporting, and file maintenance activities performed by the organisation. Under the software process employed, a System Proposal document outlining the functionality to be delivered was signed off by the client after one week. On system delivery, the client performed an acceptance test and system review. All projects satisfied the requirements of both the client, as evidenced by the reviews, and the course administrators, as indicated by the marks awarded (although marks varied over the sample). Note that failures were excluded.

A wide variety of systems was constructed over the period. A few examples should help to provide an indication of their general nature. The set included: client management systems for two dental surgeries, a law firm and an accountant; stock/inventory management systems for five utility firms and four retail stores; membership record systems for two leisure clubs; exhibit management systems for two museums; a freight scheduling system for a transport firm; and three school course and student administration systems. In total, more than seventy distinct working systems were developed and reviewed.

The systems were all of small to medium size, as illustrated by the following indicators of scale: each system included an average of eleven data entities and sixty attributes; six reports were produced on average by each system, while eleven data entry and update functions were provided. In terms of code product size, the smallest system was comprised of 309 4GL source statements, the largest contained more than 2600 statements, and the average size was approximately 1100 code statements. Thus whilst the systems were not large by any means, they were not trivial, nor purely academic exercises.

Although the developers were indeed students, they were in the final segment of their three- or four-year degrees. Almost all of those completing their degrees went on to hold development positions within three months of the completion of the projects in the sample. More importantly, the systems developed were functionally sound, providing an actual working solution to an actual client problem. The students all had an equivalent amount of previous experience with the tool and with the methodology used. Each project was overseen by a supervisor, a staff member in the Information Science department, to ensure delivery of a quality solution.

One of the positive features of the sample is the degree of commonality of several process attributes across the set. All but a few of the systems were built by groups of four developers; the same development methodology was employed in every case; all systems were implemented using the same tool, the Cognos 4GL Powerhouse; and the systems were all of the same generic class - transaction-oriented data processing and retrieval systems. This commonality is advantageous in that these factors can be considered as constant in the analysis, a condition not often encountered in software size research. When they vary, factors such as these can clearly have an impact on system size. Given that these potential contributors may be treated as constant, the degree of confidence adopted in regard to any size relationships supported by the data will consequently be greater.

2.2. Data collection

Two product sets, the system documentation and the implemented code, were examined in order to collect the appropriate data. Specification size measures were manually collected from each System Proposal. Two of the authors performed this task so as to obtain as correct a data set as possible. (Each author undertook the collection task independently, then the two data sets were compared and any discrepancies were identified and resolved.) Within each set of documents a number of measures were extracted in order to address the following questions:

- is data model size related to the size of the implemented system?
- is functional model size related to the size of the implemented system?

This approach was based on the assumption that consideration of the two dimensions of data and function could provide adequate independent indicators of system size. The measures collected were coarse, in line with one of the objectives of the study; that is, to test for the existence of size relationships using high-level, objective, and easily extracted indicators. The measures of data model size collected in this study were therefore:

1. the number of entities depicted in the entity-relationship diagram (ENT)
2. the number of relationships depicted in the entity-relationship diagram (RSHIP)
3. the number of attributes associated with the entity-relationship diagram (ATTRIB).

The functional model size measures were of a similarly coarse nature:

1. the number of menus depicted in the functional decomposition chart (MENU)
2. the number of data entry/edit screens depicted in the functional decomposition chart (EDIT)
3. the number of reports depicted in the functional decomposition chart (REPORT)
4. the number of non-menu functions depicted in the functional decomposition chart (NONMENU)
5. the total number of functions depicted in the functional decomposition chart (FDCSIZE).

(The terms in brackets after each measure are the variable names used in the presentation of results below. Functional sizes four and five are simply the totals of sizes two and three, and sizes one, two and three, respectively.)

The second product set examined was the implemented code itself. All source program files for each system were scanned automatically by a parsing program to extract the following size measure:

1. the total number of source statements in the system (SIZE).

This measure excluded blank and comment lines of code, and counted run-on lines with a continuation indicator as a single statement. The extraction of the measure was verified manually on a random selection of ten programs by one of the authors to ensure consistency and reliability. (No counting errors were identified by this check.)

Of the seventy-four systems in the total sample four were incomplete, in that full specification documents were not available. Consequently a usable data set of eight specification measures and one implementation measure was collected from seventy systems. The distribution of systems over the period was as follows: thirteen from 1991, ten from 1992, sixteen from 1993, thirteen from 1994 and eighteen from 1995. (The actual data set is provided in the appendix in the interests of further analysis.)

3. DATA ANALYSIS AND RESULTS

3.1. Descriptive statistics

General descriptive statistics for each of the variables are shown in Table 1. The descriptive indicators highlight the absence of significant skewing for all but the MENU and EDIT variables. Further analysis using boxplot distributions enabled outliers to be identified, with the frequency of outliers (O) and extreme outliers (E) shown in the right-most column of Table 1.

3.2. Correlation analysis

Given the small degree of skewing in some of the distributions, correlation tests were performed using both the Pearson and nonparametric Spearman statistics so as to identify any potentially useful (linear) relationships between the specification size variables and the implementation size measure, as well as among the specification variables themselves. The results are shown in Tables 2 and 3.

Both sets of correlation statistics provided evidence of strong significant relationships between several of the specification size variables and the implementation size measure. In particular, the relationships between the ATTRIB, NONMENU and FDCSIZE specification measures and implementation SIZE were strong even when tested with the more conservative Spearman statistic. This suggested that, for the range of system size considered here, potential predictive relationships might have been derivable. Some cross-correlation was also evident among the specification size variables themselves, suggesting that several of the measures may have in fact been assessing the same size characteristic.

3.3. Regression

Given the strong correlation identified between the two variable types a regression model to determine implementation size from specification size measures was sought. When appropriate, linear models, with their inherent simplicity and intuitive appeal, are generally preferred over more complex non-linear models - evidence of a strong linear relationship in this data set was illustrated by the significantly strong positive values for the Pearson correlation statistic (see Table 2).

Stepwise linear regression was therefore undertaken on a randomly selected set of 50 observations, with the following model being produced:

$$\text{SIZE} = -155.0 + 45.5(\text{NONMENU}) + 6.2(\text{ATTRIB})$$

This model had an associated adjusted R^2 value of 0.65, indicating that it explained 65% of the variance in implementation size. The R^2 value provides a measure of the consistency of a specific regression model. This was a particularly pleasing result, suggesting that a significant degree of implementation size variation can be attributed to (or at least predicted from) earlier product size.

A question may be raised as to the acceptability of the model, given the inclusion of two 'independent' terms that were in fact themselves related (as illustrated in the correlation matrices presented above). Models that incorporate related terms may be unstable or less easily generalised to other data sets - this implies that caution should be exercised in accepting models of this nature [4, 13]. A regression model that included just the most significant independent variable (NONMENU) was therefore also computed. The form of this model is as shown:

$$\text{SIZE} = -59.8 + 62.3(\text{NONMENU})$$

This model's associated adjusted R^2 value was 0.61, a small decrease in explanatory ability when compared to the two-variable model.

It may be asserted that both terms (NONMENU and ATTRIB) should be included. The strength of the intercorrelation may be considered as moderately strong (at 0.7 Pearson and 0.62 Spearman), but equally it could be suggested that the two variables are still measuring somewhat different aspects of the common overall characteristic of specification size. This seems plausible, given that NONMENU is a function-based measure and ATTRIB is a data-oriented measure. One technique that can assist in determining the gain associated with including extra terms in a regression model is the R^2-adequate test [16]. A subset of predictor variables is said to be R^2-adequate (at significance level α) if:

$$R^2_{\text{sub}} > 1 - (1 - R^2_{\text{full}})(1 + d_{n,k})$$

where

R^2_{sub} is the R^2 value achieved with the subset of predictors

R^2_{full} is the R^2 value achieved with the full set of predictors

$d_{n,k} = (kF_{k,\, n-k-1})/n-k-1$

where

k = number of predictor variables in the model

n = the number of observations

F = the F statistic for significance α for n,k degrees of freedom

In this case:

$k = 2$, $n = 50$, $\alpha = 0.05$

$R^2_{\text{full}} = 0.65$ (for the two-predictor model)

$R^2_{\text{sub}} = 0.61$ (for the single-predictor model)

$$\begin{aligned}
\Rightarrow \quad d_{50,2} &= (2F_{2,\,47})/50-2-1 \\
&= (2*3.2)/47 \\
d_{50,2} &= 0.136
\end{aligned}$$

$$\begin{aligned}
\Rightarrow \quad R^2_{\text{sub}} \quad > \quad & 1 - (1 - R^2_{\text{full}})(1 + d_{n,k}) \\
& 1 - (1 - 0.65)(1 + 0.136) \\
& 1 - (0.35)(1.136) \\
& 1 - 0.3976 \\
& \quad 0.6024
\end{aligned}$$

Variable	Median	Mean	Std Devn	Min	Max	Skew	Outliers?
ENT	11.0	11.6	4.55	4	26	0.78	2O
RSHIP	9.0	10.2	5.14	3	25	0.91	1O
ATTRIB	59.5	64.5	23.85	25	141	0.77	1O
MENU	5.0	5.6	2.18	4	14	2.12	4O 2E
EDIT	11.0	12.0	4.82	4	27	1.22	4O
REPORT	6.0	6.8	3.39	1	17	0.75	1O
NONMENU	17.5	18.8	6.95	10	39	1.06	3O
FDCSIZE	22.0	24.4	8.06	14	45	0.96	-
SIZE	993.5	1106.0	543.61	309	2605	1.00	2O

Table 1: Descriptive statistics for each measure

Variable	SIZE	ENT	RSHIP	ATTRIB	MENU	EDIT	REPORT	NONMENU
ENT	.5809							
RSHIP	.5465	.9513						
ATTRIB	.6772	.7163	.7018					
MENU	.3366	.3409	.3453	.2923*				
EDIT	.7509	.6819	.6077	.6965	.2761*			
REPORT	.6074	.2870*	.2802*	.4366	.4108	.4185		
NONMENU	.8162	.6123	.5576	.6953	.3914	.8967	.7772	
FDCSIZE	.7952	.6204	.5744	.6789	.6079	.8483	.7816	.9686

Table 2: Pearson correlation coefficients
(All significant at 0.01 level, except for * significant at 0.05 level)

Variable	SIZE	ENT	RSHIP	ATTRIB	MENU	EDIT	REPORT	NONMENU
ENT	.4948							
RSHIP	.4670	.9525						
ATTRIB	.6635	.6643	.6307					
MENU	.3827	.3790	.3741	.3443				
EDIT	.6677	.6834	.6382	.6709	.4818			
REPORT	.5271	.2058*	.1750**	.3537	.2989	.3631		
NONMENU	.7171	.5512	.4977	.6219	.5024	.8473	.7800	
FDCSIZE	.7227	.5616	.5236	.6123	.6758	.8251	.7422	.9632

Table 3: Spearman correlation coefficients
(All significant at 0.01 level, except for * significant at 0.05 level and ** significant at 0.1 level)

Given that the R^2 value of the single-predictor model (at 0.61) is indeed greater than the minimum threshold R^2-adequate value for the full (two-predictor) model (at 0.60), we can say that the model including just the NONMENU term is as effective in terms of consistency as the model containing both NONMENU and ATTRIB. Moreover, adopting this single-predictor model may help to overcome some of the problems referred to above in relation to including related predictive terms in a regression model. In addition, the single predictor model would seem to be more plausible due to the smaller negative intercept. Ideally one would expect a zero or small positive size associated with the implementation of a null system.

Sufficient model consistency as illustrated by the R^2 statistic is an important and desirable aspect of a regression model, but it is by no means the only sought-after characteristic. In fact, a model may very well be *consistent* but it may still be excessively inaccurate, in terms of the correspondence of individual value pairs. Indicators commonly used in software metrics data analysis to evaluate the *accuracy* of regression models are the mean magnitude of relative error (MMRE) and the threshold-oriented pred measure.

The magnitude of relative error (MRE) is a normalised measure of the discrepancy between actual values (V_A) and fitted values (V_F):

$$\text{MRE} = \text{Abs}((V_A - V_F)/V_A)$$

The mean MRE is therefore the mean value for this indicator over all observations in the sample. A lower value for MMRE generally indicates a more accurate model.

The pred measure provides an indication of overall fit for a set of data points, based on the MRE values for each data pair:

$$\text{pred}(l) = i/n$$

where l is the selected threshold value for MRE, i is the number of data pairs with MRE less than or equal to l, and n is the total number of data pairs.

As an illustration, if pred(0.30) = 0.6, then we can say that 60% of the fitted values fall within 30% of their corresponding actual values.

For this part of the study, the two regression models were applied to the validation set of twenty observations that was randomly extracted prior to model development. The relevant predictive accuracy values are presented for the models in Table 4. These results again suggest that the single variable model may be preferred in terms of obtaining a simple and effective predictive equation for implementation size. Whilst selection of the single-variable model does mean that none of the data model measures is explicitly incorporated, the fact that

NONMENU and ATTRIB are themselves correlated suggests that NONMENU alone takes some account of the data contribution. Furthermore, ATTRIB could have been used alone in the estimation model, but with some reduction in predictive power.

3.4. Residual analysis

Residual analysis, in which predictive errors are considered in relation to both the estimated and actual values, indicated a slight tendency to overestimate for smaller systems. However, residual plot examination did not show any significantly problematic trends in the error distributions, except to suggest that the models were not *fully* accounting for the variation in size (a fact already evident in the other statistical indicators of model adequacy).

4. DISCUSSION

This work shows that it is possible to predict 4GL system size at a fairly early stage in a project using simple and objective counts derived from a functional specification and data model. In particular, we found that the numbers of non-menu screens and attributes were the primary inputs for prediction systems derived from regression analysis of our data set. Such relationships are both simple and intuitive.

	MMRE	pred(30)	pred(20)	pred(10)	pred(5)
NONMENU & ATTRIB	.21	.80	.65	.25	.10
NONMENU only	.21	.75	.65	.30	.25

Table 4: Indicators of model accuracy

4.1. Outliers

A number of outlying data points were encountered in the data set. As a result, these points were examined in greater depth, to determine whether the observations were valid within the context of the sample. In particular, one observation stood out as a significant implementation size outlier with an associated MRE of 1.71 under the two-variable model, a value twice the MRE of the next most significant outlier point. This project was found to be the smallest of the entire sample at just 309 lines of source code.

On further investigation, it was found that the system had been developed using the maximum of default settings and generated code, with very little programmer adaptation to customise the functionality and user interface employed. Although admittedly *unusual*, this did not make the project *invalid* in terms of the study - thus the observation was left in. Project managers should be aware, however, that minimalist development of this nature may lead to similar outliers in their own data sets and that predictive equations may not be as effective for such systems.

4.2. Relationship to other work

The first point to note is that our results are in line with, although somewhat better (MMRE=21% compared with 36% and 53%) than, the study by Bourque and Côté [3]. This replication of results is suggestive of the general possibility of building prediction systems for 4GL software for characteristics such as size. It must, however, be stressed that we are not advocating a “single model fits all” type solution. Models must be calibrated to suit different measurement environments. The use of student projects as a basis for the analysis may also be challenged in some respects, with the suggestion that results may not generalise to industry. In terms of the specific model, this may well be the case. It is our contention, however, that with the common use of standards in specification and coding used in many mature software organisations, the development of *similar* models of this nature (predicting size from simple early measures) should be feasible and potentially rewarding.

The next point is why not use function points? There are a number of reasons why we avoided a function point solution. First, they are more complex to collect and it is not a measure that lends itself to automation since one is trying to extract meaning from informal specification documents. Second, there have been questions raised concerning the objectivity of the process [11, 14, 15]. In particular, Low and Jeffery report inter-rater discrepancies well in excess of 30%. Third, there is a problem of high correlation between the individual components of the function point count. Kitchenham and Kansala [12] describe a procedure for disaggregating function point counts and re-estimating weights by use of stepwise regression. Although significant intercorrelation was also found between the measures extracted in our approach, there was sufficient flexibility to leave one of the components out of the model without a significant loss of power, thus avoiding the problem of component interdependence. Fourth, function point analysis, in its comprehensive form, requires far greater detail than is generally available at the system proposal stage of a project, as was assessed in this study. In summary, we believe our approach of using counts that could be automatically extracted from ER models and functional specifications to be simpler, more effective and easily interpreted. Whilst it is true that the measures in this particular case were collected manually, there is greater potential (than for function point analysis) that this collection process could be completely automated. Indeed, since the data was gathered, a tool has been developed to extract the specification measures directly from CASE tool depictions.

5. CONCLUSIONS AND FURTHER WORK

To recap, we have conducted an empirical investigation based upon data collection from 70 4GL systems. This study showed that it is possible to predict the size of a 4GL implementation from metrics derived from the functional specification and ER model. Moreover, an MMRE of 21% was obtained which compares favourably with much other work. It is noteworthy that this level of accuracy was obtained without recourse to function point analysis and its attendant problems. We believe that these results will generalise to other 4GL systems, in the sense that it should be possible to develop prediction systems based upon simple measures such as number of attributes and non-menu screens. The reason for such confidence is that our study is based upon a large number of different systems and is indeed in some senses a replication of the earlier work by Bourque and Côté [3]. It would be tempting at this point to suggest that this particular set of relationships will hold for all 4GL system implementations in terms of providing effective predictive models. However, without similar empirical analysis of a sizeable sample of large systems such an assertion would be rash, particularly given past evidence of a difference in relationship form for system samples of substantially different scale [2]. Whilst this does reduce the generalisability of the actual equations, we can at least be confident that the relationships are not extrapolated out of context. Clearly, other factors including the heterogeneity of the data set and quality of data collection will also bring to bear upon the accuracy of any predictions. Unfortunately, we were not able to obtain reliable effort data for the projects we studied although this would be an interesting avenue of enquiry, to see whether project effort could be predicted using similar metrics.

Predicting implementation size at such an early stage in a software project is useful for the practitioner since it gives important insights into the effort required to develop the project (given that size is the major input to almost all effort estimation models). At a finer level of system granularity it may also be of use in the allocation of appropriate resources to particular system components, given that those of larger size are likely to require more extensive testing. Although the exact nature of the prediction systems will doubtless vary from site to site, the underlying principles remain the same. That is, developers can usefully collect simple measures derived from documents available early in the development process, for instance data models and functional specifications. From these measures it is possible to build effective prediction systems using techniques no more sophisticated than linear regression analysis.

ACKNOWLEDGEMENTS

Martin Shepperd was supported as a William Evans Visiting Fellow at the University of Otago whilst this research was being conducted.

REFERENCES

1. Albrecht, A.J. and Gaffney, J.R. Software function, source lines of code, and development effort prediction: a software science validation. *IEEE Transactions on Software Engineering 9*, 6 (1983), 639-648.
2. Boehm, B.W., *Software Engineering Economics*. Prentice-Hall: Englewood Cliffs, N.J., 1981.
3. Bourque, P. and Côté, V. An experiment in software sizing with structured analysis metrics. *Journal of Systems and Software 15* (1991), 159-172.
4. Coupal, D. and Robillard, P.N. Factor analysis of source code metrics. *Journal of Systems and Software 12* (1990), 263-269.
5. DeMarco, T. *Controlling Software Projects.* Yourdon Inc., New York NY, 1982.
6. Gray, R.H.M., Carey, B.N., McGlynn, N.A. and Pengelly, A.D. Design metrics for database systems. *BT Technology Journal 9,* 4 (1991), 69-79.
7. IFPUG. *Function point counting practices manual – release 4.0*. IFPUG, Westerville OH, 1994.
8. Ince, D.C., Shepperd, M.J., Pengelly, A. and Benwood, H. The metrification of data designs, in *Proc 3rd Annual Oregon Workshop on Software Metrics*, March 17-19, 1991, (Also reprinted in *Data Resource Management*, Summer 1992).
9. Jeffery, D.R., G.C. Low, and M. Barnes, 'A comparison of function point counting techniques', *IEEE Trans. on Softw. Eng.,* 19(5), pp529-532, 1993.
10. Kemerer, C.F., 'An empirical validation of software cost estimation models', *CACM,* 30(5), pp416-429, 1987.
11. Kemerer, C.F. Reliability of function point measurements: A field experiment. *Communications of the ACM 36,* 2 (1993), 85-97.
12. Kitchenham, B.A. and Kansala, K. Inter-item correlations among function points, in *Proc. 1st Intl. Symposium on Software Metrics*. Baltimore, MD: IEEE Computer Society Press, 1993.
13. Kitchenham, B.A. and Pickard, L.M. Towards a constructive quality model. Part II: Statistical techniques for modelling software in the ESPRIT REQUEST project. *Software Engineering Journal* (July 1987), 114-126.
14. Low, G.C. and Jeffery, D.R. Function points in the estimation and evaluation of the software process. *IEEE Transactions on Software Engineering 16*, 1 (1990), 64-71.
15. MacDonell, S.G. Comparative review of functional complexity assessment methods for effort estimation. *Software Engineering Journal* (May 1994), 107-116.
16. Neter, J., Wasserman, W. and Kutner, M.H. *Applied Linear Regression Models*. Irwin: Homewood IL, 1983.
17. Symons, C.R. *Software sizing and estimating: Mk II FPA (function point analysis).* John Wiley & Sons Ltd: Chichester, UK, 1991.
18. Verner, J. and Tate, G. Estimating size and effort in fourth-generation development. *IEEE Software 5* (1988), 15-22.

APPENDIX

The raw data is presented below in the following order:

Project	ENT	RSHIP	ATTRIB	MENU	EDIT	REPORT	NONMENU	FDCSIZE	SIZE
91A	10	6	92	4	11	8	19	23	898
91B	15	10	57	4	11	5	16	20	498
91C	7	5	33	4	7	3	10	14	510
91D	19	16	76	5	14	2	16	21	898
91E	8	6	34	4	9	9	18	22	597
91F	6	4	25	5	13	3	16	21	535
91G	14	12	83	7	20	7	27	34	2237
91H	13	11	49	4	13	12	25	29	1853
91I	10	8	34	4	8	5	13	17	555
91J	8	6	41	6	7	3	10	16	438
91K	19	18	59	11	10	3	13	24	1444
91M	15	16	48	5	13	6	19	24	649
91N	5	3	97	4	9	7	16	20	1062
92A	26	25	141	6	27	12	39	45	2324
92B	8	7	49	14	9	9	18	32	1119
92C	10	9	63	5	10	9	19	24	1302
92D	18	20	105	14	14	17	31	45	2096
92F	9	8	42	6	10	8	18	24	512
92G	16	16	72	5	12	6	18	23	1178
92H	17	16	80	7	17	5	22	29	1340
92I	13	10	96	9	17	13	30	39	1508
92J	17	16	80	9	21	8	29	38	1324
92K	6	4	51	5	11	5	16	21	1336
93A	7	5	51	4	6	4	10	14	1573
93B	17	16	97	5	25	13	38	43	2605
93C	13	11	64	6	9	6	15	21	679
93D	12	9	67	5	13	7	20	25	1831
93E	8	7	88	6	15	1	16	22	993
93F	14	12	73	5	12	7	19	24	1150
93G	8	6	71	8	8	11	19	27	1143
93H	20	16	115	6	26	8	34	40	2205
93I	11	9	53	5	12	9	21	26	916
93J	14	11	91	6	15	8	23	29	1192
93K	10	6	60	6	17	5	22	28	1112
93L	15	10	69	7	17	7	24	31	1431
93M	17	16	92	10	15	14	29	39	1367
93N	14	12	80	5	23	10	33	38	2177
93O	7	6	43	6	9	3	12	18	824
93P	13	13	65	7	16	4	20	27	719
94A	13	15	67	4	13	9	22	26	1646
94C	7	5	35	4	7	4	11	15	883
94D	24	25	114	8	18	14	32	40	2577
94E	11	12	73	6	16	10	26	32	1501
94G	10	7	48	4	9	5	14	18	649
94H	9	7	42	4	9	2	11	15	994
94I	11	8	58	6	10	10	20	26	1224
94J	12	10	57	4	11	7	18	22	1246
94K	6	4	34	4	5	5	10	14	426
94L	10	8	40	7	14	9	23	30	1263
94M	4	3	73	4	15	13	28	32	1481
94N	5	3	48	4	8	10	18	22	852
94O	8	7	32	4	8	5	13	17	443
95A	10	10	46	4	9	8	17	21	641
95B	15	17	83	4	13	4	17	21	1184
95C	11	10	46	4	12	2	14	18	591
95D	16	17	101	4	12	3	15	19	1210
95E	8	6	53	4	7	7	14	18	621
95F	7	7	30	6	7	3	10	16	632
95G	18	22	107	4	12	5	17	21	975
95H	14	13	74	4	10	3	13	17	585
95I	9	7	46	4	10	6	16	20	985
95J	8	8	57	4	7	7	14	18	731
95K	12	12	86	7	11	5	16	23	931
95L	6	4	50	4	8	4	12	16	614
95M	10	9	50	6	10	5	15	21	309
95N	9	7	54	4	8	5	13	17	928
95O	13	9	55	4	8	5	13	17	722
95P	11	11	60	4	10	3	13	17	1041
95Q	6	4	46	4	4	6	10	14	549
95R	12	11	62	8	10	8	18	26	856