

*Published in Management Science, Vol. 45, No. 6, June 1999, pp. 787-803.
Copyright 1999. Library of Congress Catalog Card No. 56-21107*

Permission to make digital/hard copy of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage, the copyright notice, the title of the publication and its date appear, and notice is given that copying is by permission of the Institute for Operations Research and the Management Sciences.

Performance Evaluation of General and Company Specific Models in Software Development Effort Estimation

21 July 1998

Katrina Maxwell*, Luk Van Wassenhove and Soumitra Dutta****

* Datamax, 7 bis bld. Foch, 77300 Fontainebleau, France

** INSEAD, Boulevard de Constance, 77300 Fontainebleau, France

Abstract

In this paper we present the results of our effort estimation analysis of a European Space Agency database consisting of 108 software development projects. We develop and evaluate simple empirical effort estimation models which include only those productivity factors found to be significant for these projects and determine if models based on a multi-company database can be successfully used to make effort estimations within a specific company. This was accomplished by developing company specific effort estimation models based on the significant productivity factors of a particular company and by comparing the results with those from general ESA models on a holdout sample of the company. To our knowledge, no other published research has yet developed and analysed software development effort estimation models in this way. Effort predictions made on a holdout sample of the individual company's projects using general models were less accurate than the company specific model. However, it is likely that in the absence of enough resources and data for a company to develop its own model, the application of general models may be more accurate than the use of guessing and intuition.

Acknowledgements

The authors would like to thank the European Space Agency for their funding of this project and Dr. Benjamin Schreiber in particular for initiating the data collection effort. They would also like to thank the data provider from company C10 for his data and feedback. In addition, they extend their thanks to the referees for their constructive comments.

You can contact Katrina D. Maxwell at: kmaxwell@datamax-france.com.

Introduction

Accurate effort and cost estimation of software applications continues to be a critical issue for software project managers. Lederer and Prasad (1992) found that two-thirds of all major software projects substantially overran their estimates. Although more than 20 software cost estimation models were already in existence as early as 1981 (Mohanty 1981), Lederer and Prasad found that, in practice, software development cost estimators still rely more heavily on their personal memory of the cost of similar projects than on other estimating processes. This technique was not associated with accurate estimates. In addition, the use of guessing and intuition were found to be highly positively correlated with the percentage of large projects overrunning their estimates. Only the use of similar past projects based on documented facts, the use of a simple arithmetic formula, and the use of established standards were found to lead to greater accuracy in software cost/effort estimates.

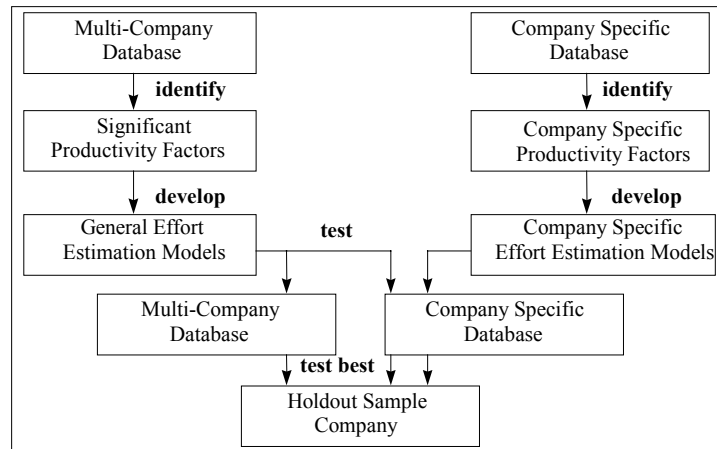
Over the past years, many effort estimation models have been developed (Walston and Felix 1977, Putnam 1978, Bailey and Basili 1981, Basili and Freburger 1981, Boehm 1981, Albrecht and Gaffney 1983, Conte et al 1986, Kemerer 1987, Matson et al 1994, to name a few); however, because of differences in data collected, project types and environmental factors among software development sites, these models are generally only valid within the organization in which they were developed (Bailey and Basili 1981). One major obstacle to the transportability of these models appears to be a lack of understanding of the factors explaining the differences in productivity among projects. As there exist literally hundreds of parameters which can affect software development effort (DeMarco 1982), and as only a few of these may effect the productivity within a given environment (Banker et al 1991, Kitchenham 1992, Mukhopadhyay and Kekre 1992), it is important to analyse the accuracy of effort estimation models based on the prior determination of the factors found to explain the productivity of

projects in a given database.

However, the reality is that organizations often do not have the resources to develop and analyze a large enough historical database of their own projects. This is why it is important to determine if effort estimation models based on the significant productivity factors of a multi-company database can be successfully used to make effort estimations within a specific company.

In this paper we present the results of our effort estimation analysis of the European Space Agency software development database which, at the time of the analysis, consisted of 108 projects from 37 companies in 8 European countries. This database is unique in that we have found no other published research which analyses such a large number of European non-MIS applications. This paper seeks to build on our previous research which investigated the productivity differences of European space, military and industrial applications (Maxwell et al 1996) by developing simple empirical effort estimation models which include only those productivity factors found to be significant in these application areas. As we previously determined that organizational differences account for most of the productivity variation of projects in the ESA dataset, it was decided to develop general ESA effort estimation models with the data from one company removed, and then to test the general ESA models on the removed company. We also develop simple company specific effort estimation models based only on the factors found to effect the productivity of this particular company. The best company specific model is then compared to models developed using the ESA dataset on a holdout sample of the company. This is illustrated in Figure 1. To our knowledge, the methodology we have employed is unique in that no other published research has yet developed and analysed software development effort estimation models on two levels: general (across companies) and specific (within company).

Figure 1. Effort Estimation Model Development and Testing on Databases



The remainder of the paper is organized as follows. An overview of prior effort estimation research is presented, followed by a description of the database and a section describing the design of the analysis. The results of the productivity and effort estimation analyses are then presented. This is followed by a concluding section which includes a summary of the results and a discussion of the limitations, as well as directions for future research.

Prior Research

Although there exists some research on the use of case-based reasoning models for software effort estimation (Mukhopadhyay et al 1992), effort estimation research has largely focused on the study of algorithmic techniques. These techniques can be categorized by the type of formula used to calculate total effort: single or multivariable, static or dynamic with regard to staffing, based on historical data or theoretical assumptions, and can use a top-down or bottom-up approach (Basili 1980). Heemstra (1992) provides a detailed overview of some recent software effort estimation models. An excellent summary of the effort estimation models available prior to 1981 can be found in Mohanty (1981).

Most effort estimation models consist of two phases: in the first phase, an estimate of the software size is made; and in the second phase, the effort and duration of the project are

estimated based on the estimated software size. As many factors besides software size have an influence on the effort and time needed to develop software, most effort estimation models also include an adjustment for a number of productivity factors (Kitchenham 1992). Models based on function point analysis (Albrecht and Gaffney 1983) are focused more on the sizing stage while others, such as COCOMO (Boehm 1981), are focused more on the productivity stage (Heemstra 1992).

In a study by Mohanty (1981), the cost of one software development project was found to vary by nearly 800% when estimated using 12 different cost/effort estimation models. As the size of the project and cost per instruction were kept constant, he concluded that the primary cause of this variation was environmental. This is because each model was developed using a specific database from a given company environment and thus embodied the productivity factors, work patterns, and management practices of that company's environment.

Some of the many factors that appear to influence software development productivity are: people factors, such as experience and team size; process factors, such as programming language and tools; product factors, such as complexity and software type; and computer factors, such as storage and timing constraints (Conte et al. 1986). The combination and interaction of all of these factors makes effort estimation very difficult

In the past, many researchers have based their models on a large number of productivity factors. In a IBM study by Walston and Felix (1977), 29 factors that were significantly correlated with productivity were found. In an analysis of data from the NASA/Goddard Space Flight Center, Bailey and Basili (1981) identified 21 productivity parameters . At ITT, Vosburgh et al. (1984) found 14 significant productivity factors. In Boehm's COCOMO model (1981), 15 software factors which had a significant impact on productivity were identified.

However, Kitchenham (1992), in her review of some of the assumptions built into

conventional software cost/effort models, found no support for the assumption that a large number of productivity factors, or size adjustment factors for models based on function points, are necessary. Few adjustment factors were found to have a significant effect on productivity within one environment. Mukhopadhyay and Kekre (1992) obtained good results in their development of an application feature based method for the early effort estimation of process control applications using only two productivity factors. The results of Banker et al (1991) suggested that a relatively small number of critical factors may explain a large amount of productivity variation at a specific site.

In addition, in models such as COCOMO, the adjustment factors are treated as if they are independent even though some are not. In Kitchenham's (1992) analysis of the COCOMO dataset, she found that some of the 15 adjustment factors used did not even significantly explain the productivity of the dataset used to develop the model. An evaluation of four algorithmic models used to estimate software effort (SLIM, COCOMO, Function Points and ESTIMACS) undertaken by Kemerer (1987) also determined that none of the models tested sufficiently reflected the underlying factors affecting productivity. Thus strong empirical evidence suggests the need for the development of a simple effort estimation model based on the prior determination of a small number of independent factors which explain the productivity variation of a given database.

The effort estimation models presented in this paper build on our recent study of the factors which influence the software development productivity of European space, military and industrial applications (Maxwell et al 1996). In this study, we found that organizational differences accounted for most of the productivity variation of projects in the ESA dataset. Some of this variation was due to the differences in the application category and programming language of projects in each company. In addition, high productivity was found in those

companies which undertook projects with low reliability requirements, low main storage constraints, low execution time constraints and which had a high use of tools and modern programming practices. We also determined that productivity decreased with increasing team size and project duration, and that programming language experience had no significant effect on productivity.

The ESA Database

In 1988, the European Space Agency, faced with the evaluation of proposals for large space programmes and their huge needs for software development, began compiling a software metrics database focusing on cost/effort estimation and productivity measures. Subsequently, the database was expanded to include military applications and custom built industrial applications such as air traffic control software and software used to control nuclear power plants. This activity is ongoing, and at the time of this analysis contained a selection of 108 completed projects from 37 companies in 8 European countries. The database collection effort consists of contacting each supplier of data on a regular basis to determine if suitable projects are nearing completion. When a completed questionnaire is received, each supplier of data is telephoned to ensure the validity and comparability of his responses. For example, we verify that the project has actually been completed and that the numbers provided are actuals and not estimates. We also verify that the definitions of kilo lines of code (KLOC), Effort and COCOMO factors have been understood, that the project is put into the correct category and each company's definition of a manmonth. Other discrepancies, such as the total duration not equaling the duration in calendar months, are also checked. In return, each data supplier receives periodic data analysis reports and diskettes of the dataset¹.

¹Confidentiality of the data is maintained. Projects are referred to by number and identifying factors, such as company and country, are withheld.

-----Table 1 about here-----

The 108 projects at the time of this analysis represented 5.51 million lines of source code (range: 2000-413000, average: 51010, median: 22300), 22 development languages or combinations of languages, and 30125 manmonths of effort (range: 7.8-4361, average: 284, median: 93). More details about the database are given in Figures 2a, 2b, 2c, 2d and Table 1.

Figure 2a.
Percentage of Projects by
Main Application Environment

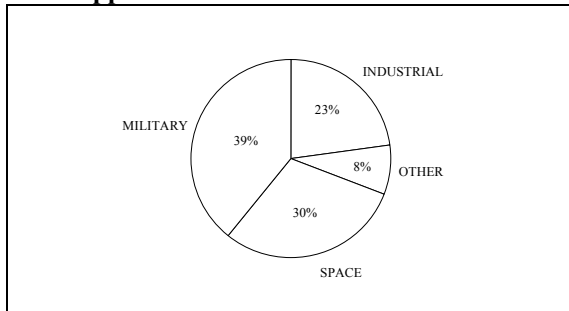


Figure 2b.
Percentage of Projects by Country

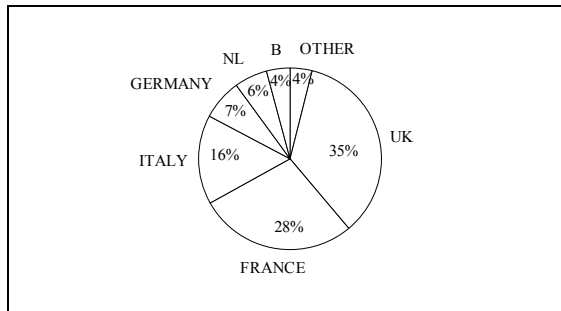


Figure 2c.
Percentage of Projects by Language

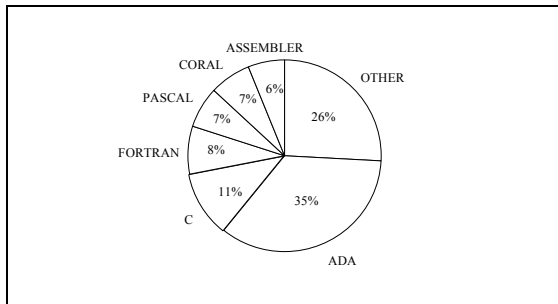
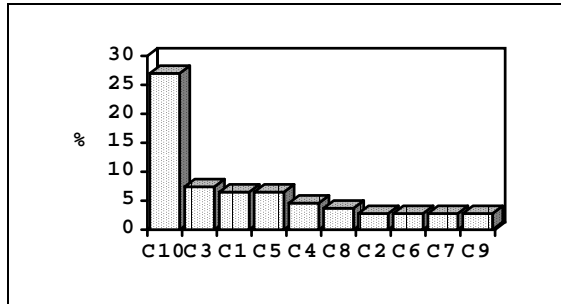


Figure 2d.
Percentage of Projects per Company
(Ten largest data suppliers)



In the ESA database, KLOC, Effort and Productivity are defined as follows:

KLOC: the amount of non-blank, non-commented delivered kilo lines of code. As the software developed in some projects consists of reused code, adaption adjustment factors (Boehm 81) were used to correct the size of the software. When no adaption adjustment factor was available, the new code size was used as a measure for the size.

Effort: The total effort was measured in manmonths and was defined as beginning at specifications delivery and ending after completion of integration and system tests. The effort value covers all directly charged labour on the project for activities during this period. All effort data has been converted to manmonths of 144 manhours per manmonth.

Productivity: In software development terms, productivity is conventionally defined as source lines of code per manmonth (LOC/MM). It is a measure of the amount of product produced per unit of human effort. Although the lines-of-code metric is the subject of much debate, the fact remains that it is considered by many organizations as a more practical productivity metric than the currently available alternatives (Boehm 1987). In a recent study by Cusumano and Kemerer (1990), the lines-of-code metric was chosen to compare productivity across a number of organizations in the US and Japan. Previous research has also shown that function points and lines-of-code tend to be highly correlated in the case of new software development (Banker et al. 1991). Until the use of function and feature point methods become common for non-MIS applications, and particularly in the domain of space, military and industrial applications, statistical analysis undertaken of large heterogenous databases will have to rely upon measuring and analysing the productivity of these types of projects using lines-of-code metrics (Maxwell et al 1996).

Design of Analysis

In our recent study of the factors which influence the software development productivity of European space, military and industrial applications (Maxwell et al 1996), we found that organizational differences accounted for most of the productivity variation of projects in the ESA dataset. Therefore, in this study, we have analysed the data on two levels: multi-company and company specific, in order to identify and compare the productivity factors found at each level and to determine if effort estimation models developed using a multi-company database can be successfully used to make effort estimations within a specific company.

As nearly 27% of the data in the ESA database was supplied by a single company (see Figure 2d), it was decided to remove this company's data in order to prevent the results from being overbiased. Thus the general ESA analysis was carried out on a subset of 79 projects. The subset of 29 projects provided by company C10 was randomly divided into two samples. A main sample consisting of 20 projects was used for the determination of the best company C10 effort estimation model, and a holdout sample consisting of 9 projects was used to test the predictive value of the model developed using the main sample, as well as general ESA models.

In the analysis, three strategies for estimating the effort of projects in the company C10 holdout sample were tested. The first strategy was for company C10 to predict effort using the general ESA model which performed best on the ESA dataset. The second strategy was for company C10 to test all the general ESA models on their main sample and to select the best performer to estimate the effort of their holdout sample. The third strategy considered was for company C10 to predict effort using their own company specific model. This is illustrated in Figure 1. The analysis was performed in five phases.

- Phase (1) The individual variables which explain the greatest amount of productivity variation of the ESA dataset and company C10 were identified and compared.
- Phase (2) The best productivity models were identified for the ESA dataset and the company C10 main sample.
- Phase (3) The best effort estimation models based on significant productivity factors were selected for the ESA dataset and the company C10 main sample.
- Phase (4) The ESA model which best estimated the effort of the company C10 main sample was selected.
- Phase (5) The best ESA model, the best predictive ESA model of the company C10 main sample, and the best company C10 model were tested on the company C10 holdout sample.

Parsimonious models were employed to examine the impact of differences in company, country, language, category, environment, start year, team size, project duration and system size, as well as the following seven COCOMO factors (Boehm 1981): required software reliability, execution time constraint, main storage constraint, virtual machine volatility, programming language experience, use of modern programming practices and use of software tools, on productivity and effort estimation. A General Linear Models procedure (SAS Institute Inc. 1989) which can analyse the variance of unbalanced data was used for this analysis. This procedure uses the method of least squares to fit general linear models. Crossed effects of class variables were taken into consideration. The analysis was performed in an unbiased way: all values were considered as being equally reliable and relationships were extracted based on the face value of the data. Both additive and multiplicative (log) models were fit to the data.

The determination of the best effort estimation model for each dataset was undertaken in two stages of analysis. In the first stage the significant factors affecting productivity were identified using a significance level of 5% (alpha=0.05).

$$\textit{Additive:} \quad \textit{Productivity} = a + b \times x_1 + c \times x_2 + \dots \quad (1)$$

$$\textit{Multiplicative:} \quad \textit{Productivity} = a \times x_1^b \times x_2^c \times \dots \quad (2)$$

where a is a constant which varies with the significant class variables and x is a productivity factor. In the second stage, the best effort estimation models based on the productivity factors found to be significant in the first stage of the analysis are determined.²

$$\textit{Additive:} \quad \textit{Effort} = a + b \times KLOC + c \times p_1 + d \times p_2 + \dots \quad (3)$$

$$\textit{Multiplicative:} \quad \textit{Effort} = a \times KLOC^b \times p_1^c \times p_2^d \times \dots \quad (4)$$

where a is a constant which varies with the class variables and p is a significant productivity factor.

In order to avoid multi-collinearity problems, any two variables with a Spearman rank correlation coefficient (Fenton 1991) exceeding + or - .75 were considered to be highly correlated and were not used in the same model. Variance inflation factors, condition indexes and variance-decomposition proportions (Belsley et al 1980) were also calculated for the effort models based on non-class variables. In addition, auxiliary regressions were undertaken on all effort models to rule out the possibility of several co-existing near dependencies among the explanatory variables.

Plots of residuals vs. fitted were examined to check for violations of least squares assumptions for all effort models. In addition, the Ramsay RESET test was used to determine if

² It should be noted that omitting the less significant variables may lead to omitted variable bias in the estimators.

there were omitted variables, and the Cook Weisberg test was used to detect heteroscedasticity for the non-class effort models. Studentized residuals and Cook's Distance were used to detect the presence of influential outliers.

The criteria for choosing the best estimation models were based on the accuracy and consistency of the effort estimates, as well as the simplicity and applicability of the model. Thus, a model based on few productivity factors was considered superior to a more complex model as long as there was not a substantial loss of accuracy and/or consistency. The accuracy of an estimation method was measured by the mean magnitude of relative error (MMRE) of the project estimates.

$$\text{MMRE} = \frac{\sum_{i=1}^N \left| \frac{\text{Actual Effort} - \text{Estimated Effort}}{\text{Actual Effort}} \right| \times 100}{N}$$

Where N is the number of observations.

A MMRE of 30% means that on average the estimates were within 30% of the actuals. The accuracy of an estimation method is inversely proportional to its MMRE. In addition, a second measure, called PRED(.25) or Prediction at Level .25, was used to measure the percentage of projects for which the predicted effort was within 25% of the actual effort. The consistency of the estimation method was measured by the correlation (CORR) between the estimates and the actuals. A consistent method should have a high positive correlation between actuals and estimates. For an in-depth discussion of accuracy and consistency refer to Mukhopadhyay and Kekre (1992).

Presentation of Results

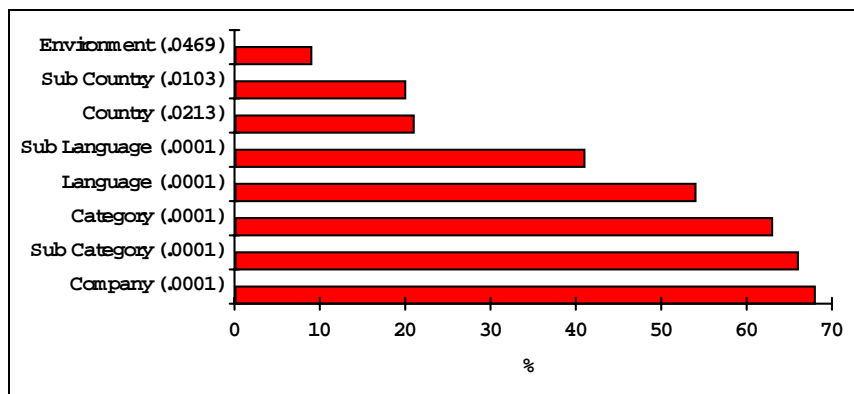
Phase I - Productivity Analysis of Individual Variables - ESA Dataset and Company C10

This phase of the analysis was concerned with determining which individual variables explain the greatest amount of productivity variation of the ESA dataset³ and the Company C10 dataset in order to highlight the fundamental differences of these two datasets. As it would not be wise to base our conclusions on the analysis of class levels that contain limited observations, analysis was undertaken on all of the data as well as for subsets⁴ that contain a sufficient number of observations at each class level. Models based on individual non-class variables were also analysed and are used to explain the differences in the class variables.

ESA Dataset

The summary of the analysis of significant individual class variables for the ESA dataset can be found in Figure 3. The level of significance is shown in parentheses. The single variable which explained the greatest amount of variance (68%) in the dataset was Company. This highlights the need for companies to establish their own software metrics database in addition to benchmarking their data against that of other companies.

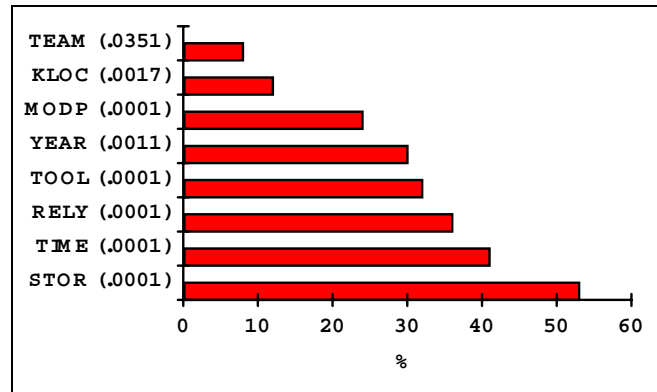
Figure 3. Percentage of productivity variance explained by significant individual class variables in ESA dataset



³In this paper, the ESA dataset refers to the ESA database with the data from company C10 removed

⁴ Referred to as sub in tables and figures.

Figure 4. Percentage of productivity variance explained by significant individual non-class variables in ESA dataset



The summary of the productivity analysis of the significant individual non-class variables can be found in Figure 4. The level of significance is shown in parentheses. Productivity was found to significantly decrease with increasing team size, and to significantly increase over time (start YEAR of project) and with increasing system size. Productivity was also found to significantly increase with increasing use of tools and modern programming practices, and with decreasing storage constraints, time constraints and required reliability. The difference in productivity between categories and languages due to these five factors may be visualized in the kivi diagrams (Figures 5 and 6). The scales have been chosen to represent a productivity increase as the shape expands. For example, in Figure 5 it can be seen that the On Board Category has a lower productivity than the Ground Support Equipment Category and that this can be partially explained by the higher time constraints, storage constraints and reliability requirements, and the lower use of tools and modern programming practices in the On Board Category. No significant relationships were found between MODP, TOOL, STOR, TIME and RELY and the start YEAR of the project. This result is probably due to the effect of combining data from different companies.

Figure 5. Median scores of productivity factors by highest and lowest productivity categories in ESA dataset

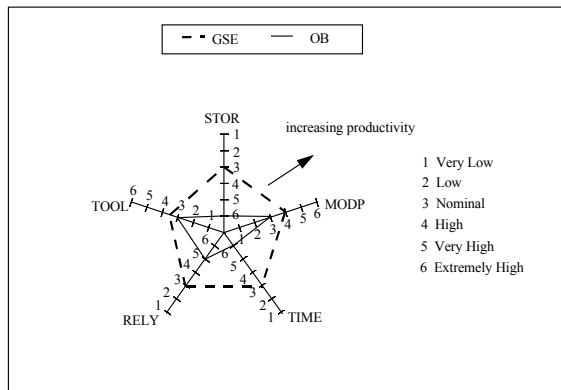
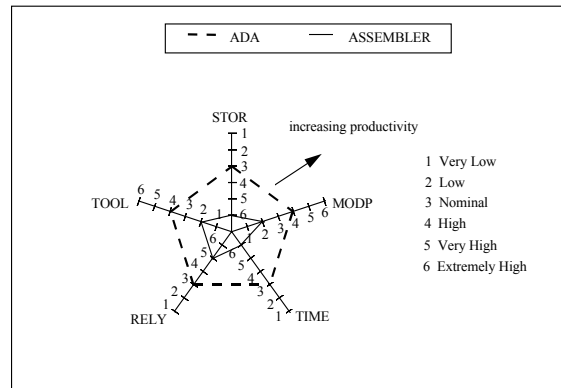


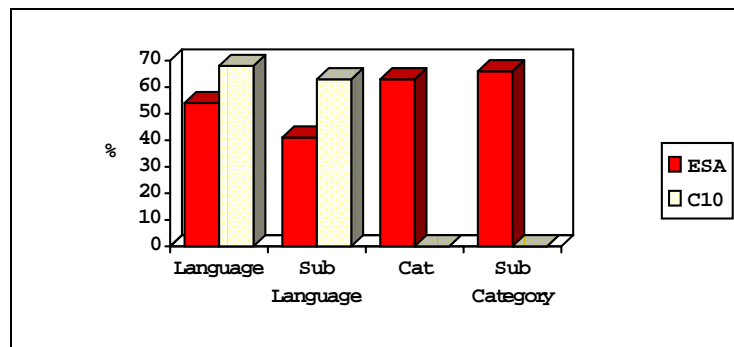
Figure 6. Scores of productivity factors by highest and lowest productivity languages in ESA dataset



Company C10

The percentage of productivity variance explained by individual class variables for company C10, compared with the ESA dataset, is shown in Figure 7. Limiting Language to the subset of 3 languages used in 3 or more projects caused the amount of variance explained by language for company C10 to decrease from 68% to 63%.

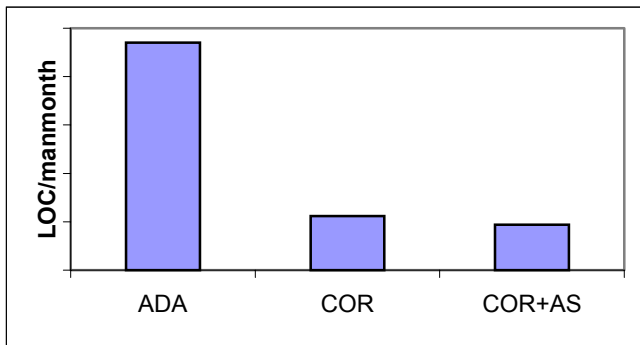
Figure 7. Percentage of productivity variance explained by individual class variables



The mean productivity of each language for company C10 and the ESA dataset are presented in Figures 8 and 9. Ada is shown to be a high productivity language and Coral (COR) and Assembler (AS) low productivity languages for both datasets. No variation was explained by the different categories of projects in company C10, even when the subset of 3 categories with 4 or

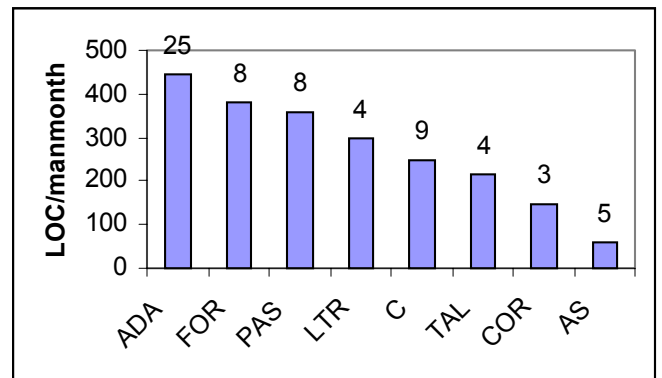
more observations was considered. This may be because most of the projects undertaken by company C10, and the entire category subset, were in the medium productivity categories of GRD, RT and SIM (Maxwell et al 1996). The mean productivity of each category for company C10 and the ESA dataset are shown in Figures 10 and 11. The Ground Control category has a lower productivity for company C10 than for the ESA dataset.

Figure 8. Mean productivity by language* language Company C10**



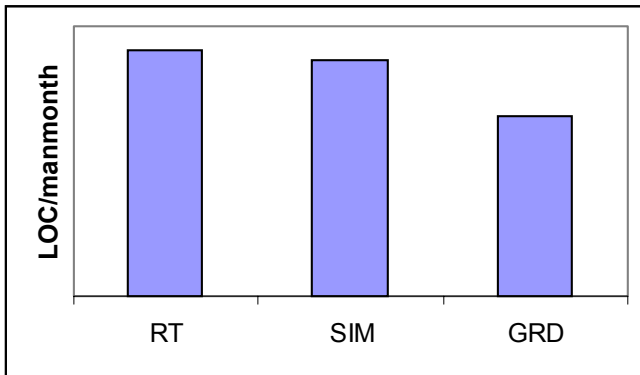
*Values removed to protect data confidentiality

Figure 9. Mean productivity by ESA dataset



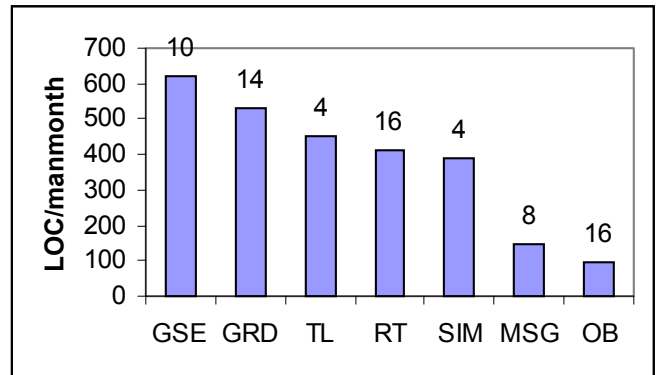
**Number of observations shown above bar

Figure 10. Mean productivity by category* Company C10



*Values removed to protect data confidentiality

Figure 11. Mean productivity by category ESA dataset**



**Number of observations shown above bar

The percentage of productivity variance explained by individual non-class variables for company C10 and the ESA dataset are presented in Figures 12 and 13. It can be seen that the factors which explain the productivity variation of company C10 differ considerably from the ESA dataset. Productivity was found to significantly decrease with increasing project duration and

team size, and to significantly increase over time for company C10. Company C10 productivity was also found to significantly increase with increasing use of modern programming practices and tools, and with decreasing storage constraints and virtual machine volatility. The non-significance of required software reliability is due to its not varying much in the dataset.

Figure 12. Percentage of productivity variance explained by individual continuous variables

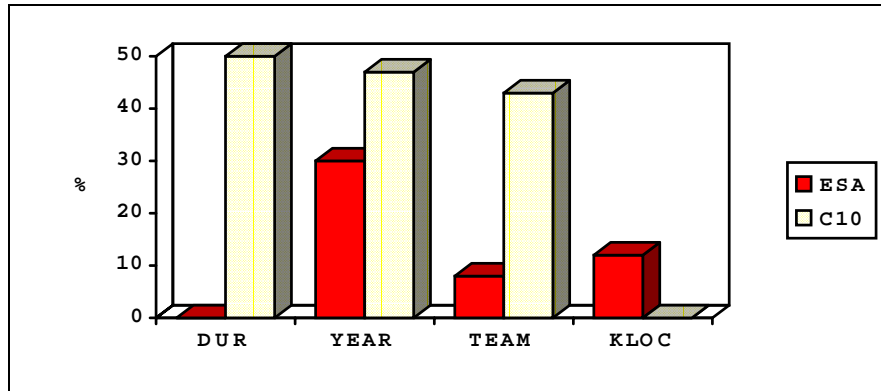
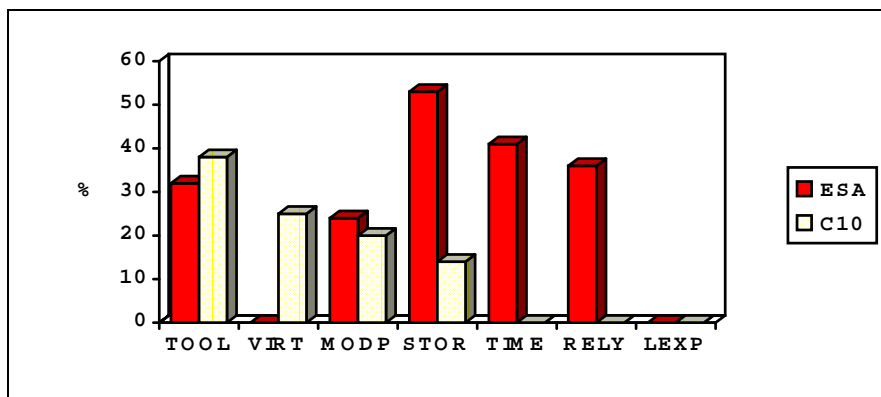


Figure 13. Percentage of productivity variance explained by 7 COCOMO factors



The difference in productivity between languages and categories due to the four significant COCOMO factors may be visualized in the following kiviatt diagrams (Figures 14 and 15). The median scores of the significant productivity factors MODP, TOOL, VIRT and STOR for the subset of 3 languages for which there were a sufficient number of observations is shown in Figure 14. It can be seen that ADA projects have a higher productivity than Coral and Coral+Assembler projects, and that this is partially due to the lower storage constraints and

virtual machine volatility, and higher use of tools and modern programming practices of the ADA projects. Figure 15 shows that there is not much difference in the scores of the four significant COCOMO factors across categories. This ties in with the fact that breakdown by project category was not found to significantly affect the productivity of company C10. This is also shown by the similarity of the median productivity factor scores for the three categories (Figure 10). Significant relationships were also found between MODP, TOOL, VIRT and STOR and time. Modern programming practices and tool use have been increasing over time, while storage constraints and virtual machine volatility have been decreasing over time for company C10.

Figure 14. Scores of productivity factors by language - Company C10

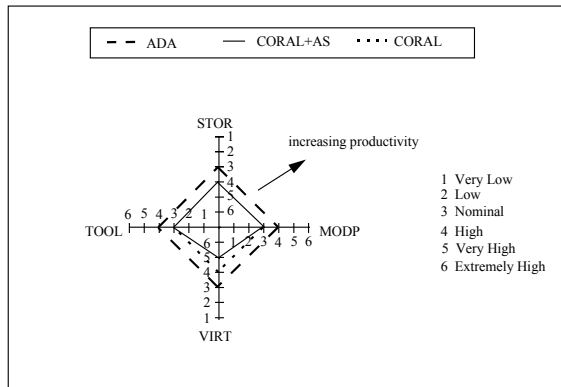
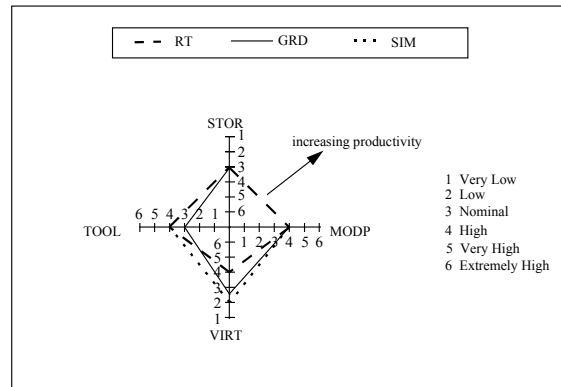


Figure 15. Scores of productivity factors by category (RT, GRD, SIM) - Company C10



Phase II - Best Productivity Models

In the second phase of the analysis, the best productivity models based on combinations of non-class and class variables were identified. Multi-variable models were run to determine what combinations of class variables, the system size (KLOC), the start year, and the seven COCOMO factors could explain the most variation in productivity. Improvement in some of the models was obtained by including the team size or duration. However, as the purpose of this

productivity analysis was to identify factors that can be used to predict effort, these productivity models are not presented⁵. In the following tables, NOBS refers to the number of observations, ROOT MSE is the root mean squared error of the model, and MODEL(significance) describes the type of model, additive or log, and its significance.

ESA Dataset

The variables MODP and TOOL were found to be significantly correlated at the 0.81 level and thus were not considered together as independent variables in the multivariate regression.

Table 2. Summary of productivity analysis of best mixed models for ESA dataset

VARIABLES	NOBS	VARIANCE EXPLAINED	ROOT MSE	MODEL (significance)
subset Category* subset Country, RELY, TOOL	54	96%	0.26	Log(.0001)
subset Category* subset Country RELY, MODP	53	95%	0.29	Log(.0001)
subset Language* subset Category RELY	60	90%	0.39	Log(.0001)
subset Category, STOR, RELY, TOOL	55	89%	0.35	Log(.0001)
subset Category STOR, RELY, MODP	55	89%	0.35	Log(.0001)
STOR, RELY, TOOL	59	70%	0.52	Log(.0001)
STOR, RELY, MODP	59	70%	0.53	Log(.0001)

** Signifies crossed effect of class variables*

The 2-class productivity models based on Country (see Table 2) do not make much sense because the productivity values for each country are based on too few observations. The best 2-class model is based on Language, Category and reliability requirements. This model explains 90% of the productivity variance and is significant at the .0001 level. Thus, in this database, companies should only compare the productivity of their projects with those in the same language and category, and with the same reliability requirements. As the objective of this analysis was to develop simple effort estimation models based on productivity factors which can

⁵Effort being by definition a function of team size and duration.

be easily applied to any company, the best productivity models based on 1 class and no class variables are also shown. It can be seen that the one-class models, which explain 89% of the productivity variance and are significant at the .0001 level, are nearly as good as the two-class model.

Company C10 main sample

The variables YEAR and TOOL were found to be significantly correlated at the 0.81 level and thus were not considered together as independent variables in the multivariate regression. Analysis of Variance Models based on combinations of class variables were not considered as categorising the data by two class variables led to too many cells with only one observation. The best company specific model was based on language and the start year of the project. This model explained 70% of the variance and was significant at the .0060 level. During the time frame considered, start year was an important productivity factor for this company because it combined the effects of modern programming practices and tool use, which had been increasing over time, and storage constraints and virtual machine volatility, which had been decreasing over time. The best productivity model for company C10 is given in Table 3.

Table 3. Best productivity model for company C10 main sample

VARIABLES	NOBS	VARIANCE EXPLAINED	ROOT MSE	MODEL (significance)
subset Language, YEAR	14	70%	0.53	Log (.0060)

Phase III - Best Effort Estimation Models

In the third phase of the analysis, the best effort estimation models based on significant productivity factors were identified. Effort estimation models based on the duration of the project or the maximum team size were rejected as not being useful to predict the effort of future projects.

ESA Dataset

The best effort prediction model was found to be based on the size of the project with the addition of the main factors found to effect productivity: application category, language and software reliability requirements (see Table 4).

Table 4. ESA effort prediction models on ESA dataset

VARIABLES	NOBS	R**2	ROOT MSE	MODEL (sign.)	MMRE	CORR	PRED(.25)
subset Category* subset Language, KLOC, RELY⁶	60	0.95	0.39	Log(.0001)	20%	0.95	67%
subset Category, KLOC, STOR, RELY, MODP ⁷	55	0.93	0.35	Log(.0001)	26%	0.94	62%
subset Category, KLOC, STOR, RELY, TOOL ⁸	55	0.93	0.35	Log(.0001)	26%	0.93	60%
KLOC, STOR, RELY, MODP ⁹	59	0.82	0.52	Log(.0001)	40%	0.95	51%
KLOC, STOR, RELY, TOOL ¹⁰	59	0.82	0.51	Log(.0001)	41%	0.94	46%
KLOC	77	0.50	0.85	Log(.0001)	79%	0.74	22%

This model has an R-squared of 0.95 and is significant at the .0001 level. The average estimation error was 20% and 67% of the estimations were within 25% of the actuals. The correlation coefficient of 0.95 shows that the model is very consistent. The five best ESA models presented are of three types: 2-class, 1-class and no-class in order to enable each company to determine which general ESA model best fits their data. It can be seen that the 1-class models, with an R-squared of 0.93 and a significance level of .0001, are nearly as accurate

⁶ Influential outliers: no outlier is influential; however, many observations are influential. This is due to the small number of observations in each category/language cell. No evidence of multi-collinearity.

⁷ Two influential outliers in the TOOL category detected. No evidence of multi-collinearity.

⁸ Two influential outliers in the TOOL category detected. No evidence of multi-collinearity.

⁹ Multi-collinearity test results: Mean VIF 1.28, Condition Index 21; Influential outliers: none with overly strong influence (DFBETA's < 1).

¹⁰ Multi-collinearity test results: Mean VIF 1.35, Condition Index 24 Influential outliers: none with overly strong influence (DFBETA's < 1).

as the 2-class models. The effort estimation equations for each model can be found in Appendix 1. A model based on KLOC only is shown for comparison.

Company C10 main sample

The best company specific effort prediction model was based on the size of the project with the addition of the main factors found to effect productivity: development language and start year of the project (see Table 5). This model had an R-squared of 0.92 and was significant at the .0001 level. The average estimation error was 41% and although only 36% of the estimations were within 25% of the actuals, 50% were within 28% of the actuals. The correlation coefficient of 0.99 shows that the model is very consistent. This model had one outlier which on further examination turned out to be a short Ada project with very high productivity. Calculation of Cook’s Distance (highest=.16) showed no projects had a high influence. No evidence of multicollinearity was found. A company C10 model based on KLOC alone is shown for comparison.

Table 5. Best C10 effort prediction models developed for company C10 main sample

VARIABLES	NOBS	R**2	ROOT MSE	MODEL (sign.)	MMRE	CORR	PRED(.25)
KLOC, subset Language, YEAR ¹¹	14	0.92	0.56	Log (.0001)	41%	0.99	36% 50%= PRED(.28)
KLOC	20	0.72	0.86	Log(.0001)	82%	0.94	20%

Phase IV - Best ESA Model on Company C10 main sample

In the fourth phase of the analysis, general effort estimation models developed using the ESA dataset in Phase III were used to predict effort for the company C10 main sample. Although none of these models was extremely accurate, it is interesting to note that the general ESA model which best predicted the company C10 main sample was the general ESA model which performed best on the ESA dataset. However, as this result may differ among companies, it is

¹¹ For model based on KLOC and YEAR only: Mean VIF=1.00, CI=285, R²=0.0011 (not sign.), correlation between KLOC and YEAR = -0.08

important to check the performance of all general models in order to select the best model for each company. This ESA 2-class model is based on category, language, and required software reliability. The average estimation error was 50% and 27% of the estimations were within 25% of the actuals. The correlation coefficient of 0.83 shows that the model is reasonably consistent. These models are presented in Table 6.

Table 6. Results of predicting company C10 main sample effort with general effort estimation models

MODEL	VARIABLES	NOBS	MMRE	CORR	PRED(.25)
ESA 2-class model	subset Category* subset Language, KLOC, RELY	11	50%	0.83	27%
ESA no class model	KLOC, RELY, STOR, MODP	18	53%	0.96	17%
ESA KLOC model	KLOC	20	64%	0.89	20%
ESA 1-class model	subset Category, KLOC, STOR, RELY, TOOL	18	60%	0.94	11%

Phase V - Results of Three Effort Estimation Strategies on Company C10 holdout sample

Three strategies for estimating the effort of the company C10 holdout sample were tested in this phase of the analysis (see section on design of analysis and Table 7). Of the 9 projects in the company C10 holdout sample, only 4 could be estimated with the best general ESA models, and 6 with the company C10 model. This is because the company was developing a project in a category or language for which no comparable data existed in the dataset considered. The best predictive model of the effort required for projects in the company C10 holdout sample was the model developed using company C10 data. This model is not yet accurate enough to be considered a very good predictive model given that the mean magnitude of relative error is 65%; however, it should be stressed that the company specific model developed in this study was based only on significant productivity factors common to the ESA dataset. Accuracy of this model could certainly be further improved through the development of effort estimation models taking into account all of the factors which may effect the productivity in the company. Although

the holdout sample contained 1/3rd of the company's data, the small number of projects severely restricted this analysis, especially as the models developed could not be applied to all projects. This resulted in the measures of accuracy being very sensitive to the overestimation of individual projects. For example, the best C10 model has a Median MRE of only 36%; the MMRE of 65% is due to the overestimation of one project.

Table 7. Results of predicting company C10 holdout sample effort with best effort estimation models

MODEL	VARIABLES	NOBS	MMRE	CORR	PRED(.25)
Best ESA on ESA dataset	subset Category* subset Language, KLOC, RELY	4	36%	.16 not significant	25%
Best ESA on C10 main sample	subset Category* subset Language, KLOC, RELY	4	36%	.16 not significant	25%
Best C10 on C10 main sample	subset Language, KLOC, YEAR	6	65%	.96	50%

Conclusions

The effective management of software development continues to be an important challenge for managers. There are two major factors contributing to this challenge. Firstly, the complexity and criticality of software within industry is high and continues to grow significantly every year as software becomes an increasingly important component in many products. The amount of software code in most consumer products is currently estimated to be doubling every two years (Gibbs 1994). Secondly, despite significant progress in computing over the last decades, software development continues to be perceived as more of an art than a science in most organizations. Surveys undertaken by the Software Engineering Institute at Carnegie Mellon University show that the vast majority of organizations (over 75%) are stuck at level 1 of their Capability Maturity Model (Gibbs 1994). At level 1 of this five level software process maturity model, the organization is characterised as having an ad hoc, or possibly chaotic, process (Marciniak 1994). It is not uncommon for managers to use guessing and intuition to estimate

different aspects of a software development project. However, this practice has been found to be highly positively correlated with the percentage of large projects overrunning their estimates (Lederer and Prasad 1992).

In order to address this challenge, the objectives of this paper were firstly, to develop and evaluate simple general effort estimation models for European space, military and industrial software applications based only on those factors found to significantly impact the productivity of these types of projects; and secondly, to determine if models based on a multi-company database can be successfully used to make effort estimations within a specific company. The latter was accomplished by developing a simple company specific effort estimation model based only on those factors, common to the ESA dataset, which had a significant impact on this company's productivity; and by comparing the accuracy and consistency of the effort estimates of the company specific model to the general ESA models on a holdout sample of the company. The development and testing of the effort estimation models is illustrated in Figure 1. To our knowledge, this methodology is unique in that no other published research has yet developed and analysed software development effort estimation models on two levels: general (across companies) and specific (within company). Our study confirms the hypothesis that only a small number of productivity factors are necessary to develop a fairly accurate effort estimation model within one environment. We found that the best general effort estimation models were based on the size of the project and the main factors found to effect the productivity of the ESA dataset: application category, language, required software reliability, main storage constraint, and the use of modern programming practices or software tools. The best company specific effort estimation model was based on the two major factors, common to the ESA dataset, which explained the productivity of the individual company: language and the start year of the project. However, it appears that the predictive powers of these models on a holdout sample of similar projects was

limited. This may be because productivity factors other than the ones considered in this study would better predict effort.

Also of managerial interest is the fact that we have determined that effort predictions made for the individual company's projects using the general models were less accurate than the company specific model. The management implications of these findings are that it is best for organizations to build up their own software development database based on the factors which they believe are responsible for the productivity variation of their projects. When this database contains about 10 projects (DeMarco 1982, Shepperd and Schofield, 1996), it should be analysed to determine which factors are really causing the productivity variation. An effort estimation model can then be developed using these significant productivity factors. Further data collection can then be limited to a smaller number of productivity factors, keeping in mind that the significance of productivity factors may vary over time in a particular company. If the creation of such a database is not possible, the application of general models may be more accurate than the use of guessing and intuition.

It should also again be stressed that the company specific model developed in this study was based only on significant productivity factors common to the ESA dataset. Accuracy of this model could certainly be further improved through the development of effort estimation models taking into account all of the factors which may effect the productivity in the company.

We now address some of the limitations of our study. The relevance of any analysis is greatly limited by the quality of the data available. One of the weaknesses of the ESA database is that no data on quality (such as errors, customer satisfaction, etc...) were collected and thus the quality of the projects cannot be assessed. The analysis would also have been much more meaningful had metrics concerning management factors been collected. Before any data collection effort, the hypotheses to be tested should be formulated, terms should be precisely

defined, and care should be taken to ensure that the scales used to collect the data support statistical analysis. The ESA questionnaire has now been modified to these criteria and includes quality and management factors.

Another limitation, shared by any multi-company database, is that it is extremely difficult to ensure that each company understands each question in the same way. We can attempt to validate answers in a telephone conversation, but this will never be as exact as the data that could be obtained in a specific company where one person is in charge of measuring and collecting data for software development projects. However, if an organization does not have the resources to develop and analyse a database of their own projects, a general model is the next best option. In particular, a general model built using data from a consortium of companies with similar projects. The analysis and validation of this confidential data by an independent third party can lead to greater knowledge for all contributing companies than would otherwise be the case.

We conclude with some directions for future research. As more companies contribute to the ESA database, we will assess the performance of the general model on companies other than C10 in order to determine which types of companies can benefit from contributing to a multi-company database. A continuous input of new projects is needed to keep the database up-to-date and to allow the analysis of data over time to determine if a significant learning effect has occurred, and to check whether models built using early project data can be applied to more recent projects. We also hope to develop an improved company specific effort estimation model using additional data from company C10. We are now also applying our two-level methodology of analysing general and company specific models to an extensive multi-company database consisting of MIS projects.

References

- Albrecht, A.J. and J.E. Gaffney, Jr., "Software Function, Source Lines of Code, and Development Effort Prediction: A Software Science Validation", *IEEE Transactions on Software Engineering*, Vol. SE-9, No. 6, (Nov. 1983), 639-648.
- Bailey, J.W. and V.R. Basili, "A Meta-Model for Software Development Resource Expenditures", *Proceedings of the Fifth International Conference on Software Engineering*, San Diego, CA, (1981), 50-60.
- Basili, V.R., "Resource Models", *Tutorial on Models and Metrics for Software Management and Engineering*, IEEE Catalog No. EHO167-7, (1980), 4-9.
- Basili, V.R. and K. Freburger, "Programming Measurement and Estimation in the Software Engineering Laboratory", *The Journal of Systems and Software* 2, (1981), 47-57.
- Banker, R.D., S.M. Datar and C.F. Kemerer, "A Model to Evaluate Variables Impacting the Productivity of Software Maintenance Projects", *Management Science*, Vol. 37, No. 1, (January 1991), 1-18.
- Belsley, D.A., E. Kuh and R.E. Welsch, *Regression Diagnostics*, John Wiley & Sons, Inc., New York, 1980.
- Boehm, B.W., *Software Engineering Economics*, Prentice-Hall, Inc., Englewood Cliffs, N.J., 1981.
- Boehm, B.W., "Improving Software Productivity", *IEEE Computer*, vol. 20, (September 1987), 43-57.
- Conte, S.D., H.E. Dunsmore and V.Y. Shen, *Software Engineering Metrics and Models*, The Benjamin/Cummings Publishing Company, Inc., Menlo Park, California, 1986.
- Cusumano, M.A. and C.F. Kemerer, "A Quantitative Analysis of U.S. and Japanese Practice and Performance in Software Development", *Management Science*, Vol. 36, No. 11, (November 1990), 1384-1406.

- DeMarco, T., *Controlling Software Projects*, Yourdon Press, New York, N.Y., 1982.
- Fenton, N.E., *Software Metrics: A Rigorous Approach*, Chapman and Hall, London, 1991.
- Gibbs, W.W., "Software's Chronic Crisis", *Scientific American*, (September 1994), 72-81.
- Heemstra, F.J., "Software Cost Estimation", *Information and Software Technology*, Vol. 34, No. 10, (Oct. 1992), 627-633.
- Kemerer, C.F., "An Empirical Validation of Software Cost Estimation Models", *Communications of the ACM*, Vol. 30, No. 5, (May 1987), 416-429.
- Kitchenham, B.A., "Empirical studies of assumptions that underlie software cost-estimation models", *Information and Software Technology*, Vol. 34, No. 4, (April 1992), 211-218.
- Lederer, A.L. and J. Prasad, "Nine Management Guidelines for Better Cost Estimating", *Communications of the ACM*, Vol. 33, No. 2, (Feb. 1992), 51-59.
- Matson, J.E., B.E. Barrett and J.M. Mellichamp, "Software Development Cost Estimation Using Function Points", *IEEE Transactions on Software Engineering*, Vol. 20, No. 4, (April 1994), 275-287.
- Marciniak, J.J., *Encyclopedia of Software Engineering, Vol. 2*, John Wiley & Sons, Inc., New York, 1994.
- Maxwell, K., L. Van Wassenhove and S. Dutta, "Software Development Productivity of European Space, Military and Industrial Applications", *IEEE Transactions on Software Engineering*, (Oct. 1996), 706-718.
- Mohanty, S.N., "Software Cost Estimation: Present and Future", *Software-Practice and Experience*, Vol. 11, (1981), 103-121.

- Mukhopadhyay, T. and S. Kekre, "Software Effort Models for Early Estimation of Process Control Applications", *IEEE Transactions on Software Engineering*, Vol. 18, No. 10, (Oct. 1992), 915-924.
- Mukhopadhyay, T., S.S. Vicinanza and M.J. Prietula, "Examining the Feasibility of a Case-Based Reasoning Model for Software Effort Estimation", *MIS Quarterly*, (June 1992), 155-171.
- Putnam, L.H., "A General Empirical Solution to the Macro Software Sizing and Estimating Problem", *IEEE Transactions on Software Engineering*, Vol. SE-4, No. 4, (July 1978), 345-361.
- SAS Institute Inc., SAS/STAT User's Guide, version 6, fourth edition, vol. 2, SAS Inst. Inc., Cary, N. Carolina, 1989.
- Shepperd, M. and C. Schofield, "Effort Estimation by Analogy: A Case Study", *Proceedings of the 7th European Software Control and Metrics Conference*, (1996), 190-196.
- Vosburgh, J, B. Curtis, R. Wolverson, B. Albert, H. Malec, S. Hoben and Y. Liu, "Productivity Factors and Programming Environments", *Proceedings of the Seventh International Conference on Software Engineering*, (1984), 143-152.
- Walston, C.E. and C.P. Felix, "A method of programming measurement and estimation", *IBM Systems Journal* 16, No. 1, (1977), 54-73.

Table 1. Variables in ESA dataset

Variable Name	Type	Values	Definition
LANG	class	22 ¹²	Application Programming Language
	8 main languages	ADA	Ada
		PAS	Pascal
		FOR	Fortran
		LTR	LTR
		C	C
		TAL	TAL
		COR	Coral
		AS	Assembler
ENV	class	3	Environment (Space, Military, Industry)
COMPANY	class	37	Company where project was developed
COUNTRY	class	8	Country where project was developed
CATEGORY	class	8	ESA Classification
		OB :	On Board
		MSG :	Message Switching
		RT :	Real Time
		GSE :	Ground Support Equipment
		SIM :	Simulators
		GRD :	Ground Control
		TL:	Tool
		OTH :	Other
TEAM	continuous		Maximum size of implementation team
DUR	continuous		Duration of project in months
KLOC	continuous		Kilo Lines of Code
YEAR	continuous		Start Year of Project
			COCOMO factors (Boehm 1981) ¹³
RELY	ordinal	1-6	Required Software Reliability
TIME	ordinal	1-6	Execution Time Constraint
STOR	ordinal	1-6	Main Storage Constraint
VIRT	ordinal	1-6	Virtual Machine Volatility
LEXP	ordinal	1-6	Programming Language Experience
MODP	ordinal	1-6	Use of Modern Programming Practices
TOOL	ordinal	1-6	Use of Software Tools

¹²Number of classes.

¹³ 1 is extremely low, 2:low, 3: nominal, 4: high, 5: very high, 6: extremely high

Appendix 1
General ESA Effort Estimation Equations

ESA 2-class model

$$Effort = 1.6 \times LC \times KLOC^{0.92} \times RELY^{1.15}$$

where LC is the Language/Category Multiplier estimated from the model (see Table 8). In the event that there is no value of LC given for the needed Language and Category the 1-class or no-class ESA models should be used.

Table 8. Value of multiplier LC for each language and category

Lang / Category	GSE	GRD	TL	RT	SIM	MSG	OB
Ada	0.49	0.28	0.52	0.49	0.49	0.79	1.10
Fortran	0.57	0.35		0.63	0.47		
Pascal	0.44	0.17	0.81	0.41		0.69	1.86
LTR		0.38		0.84			1.23
C		0.54	1.40		1.02		1.16
TAL		0.15				1	
Coral				0.73		1.22	
Assembler							2.04

ESA 1-class models

$$Effort = 1.5 \times CAT1 \times KLOC^{1.02} \times STOR^{0.73} \times RELY^{0.87} \times MODP^{-0.73}$$

$$Effort = 2.1 \times CAT2 \times KLOC^{1.00} \times STOR^{0.67} \times RELY^{0.84} \times TOOL^{-0.88}$$

where CAT1 and CAT2 are the Category Multipliers estimated from the model and defined in Table 9.

Table 9. Value of CAT1 and CAT2 for each Category.

Category	CAT1	CAT2
GRD	0.35	0.38
GSE	0.59	0.57
MSG	1.33	1.46
OB	1.32	1.27
RT	0.67	0.66
SIM	0.71	0.71
TOOL	1.00	1.00

ESA no-class models

$$Effort = 1.4 \times KLOC^{0.88} \times RELY^{0.83} \times STOR^{1.23} \times MODP^{-0.93}$$

$$Effort = 2.0 \times KLOC^{0.88} \times RELY^{0.89} \times STOR^{1.06} \times TOOL^{-1.11}$$