# FINDING RELATIONSHIPS BETWEEN EFFORT AND OTHER VARIABLES IN THE SEL

Victor R. Basili and N. Monina Panlilio-Yap

Department of Computer Science
University of Maryland
College Park MD 20742

## ABSTRACT

Estimating the amount of effort required for a software development project is one of the major aspects of resource estimation for that project. In this study, the relationship between effort and other variables for 23 Software Engineering Laboratory projects that were developed for NASA/Goddard Space Flight Center was examined. These variables fell into two categories: those which can be determined in the early stages of project development and may therefore be useful in a baseline equation for predicting effort in future projects, and those which can be used mainly to characterize or evaluate effort requirements and thus enhance our understanding of the software development process in this environment. Some results of the analyses are presented in this paper.

## 1. INTRODUCTION

The estimation of resources required in the development of a software project is an issue of importance to managers. The development of useful models and equations for predicting the cost of a project is one of the major goals of software engineering. One of the ways of measuring cost is to measure the amount of effort and resources required for a project.

Several studies on measures of effort have been made and two basic approaches have been taken in these studies. Wolverton [1], Putnam [2], and Boehm [3], among others, have developed generalized models which are then parameterized for a given environment in order to predict effort. The models are based upon data from at least one environment which is hoped to be typical or representative. Walston and Felix [4], Jeffery and Lawrence [5], Basili and Freburger [6], Boydston [7], etc., have collected data from several projects in a given environment and used these data to build models for characterizing or predicting effort in that environment, as was done in the Software Engineering Laboratory. Because of the differences in the environments, the types of projects and the data collected, Bailey and Basili [8] have suggested that even generalized models are not necessarily transportable to other environments where a different set of factors come into play in different degrees.

Bailey and Basili [8] have proposed a method for generating a resource estimation model for a particular organization based on data collected in that environment. These data would capture environmental factors and differences among projects which may have some impact on the software development process. The basic approach is as follows: A background equation is computed. The factors that could possibly explain the difference between the actual effort and the effort predicted by the background equation for the available data are analyzed. The model is then used to predict the effort required for a new project. The approach requires a local data base. If such a data base is not available then clearly one of the generalized models is best.

It has been suggested by Boehm [9] that lines of code is not necessarily the best predictor for effort. In a study conducted at the IBM Santa Teresa Laboratory, Boydston [7] has found that the number of modified modules is very strong statistically and is superior to lines of code as a single variable determinator of effort. Thus the SEL summary statistics file was searched for other variables that might be better to use, especially in a baseline equation.

This paper presents some of the results of exploratory analysis on data collected in the Software Engineering Laboratory, a joint effort of NASA/Goddard Space Flight Center, Computer Sciences Corporation and the University of Maryland, which seeks to characterize and evaluate various models, metrics and software engineering practices to improve our understanding and management of both the software development process and the product. An attempt is made to find a model for effort as a function of various variables. This study also reexamines some of the relationships derived in an earlier study in the SEL by Basili and Freburger [6] based on fewer data points. The complete set of results from this study may be found in [10].

## 2. BACKGROUND

Data were collected in the Software Engineering Laboratory from ground support software projects at NASA Goddard Space Flight Center. These projects were designed for similar applications. The code is written mostly in FORTRAN except for a small percentage written in macro assembler. Three sets of data were used in this study. One set (DS1) contained 23 data points. The other two sets were subsets of this. One of them (DS2) contained projects under 50 K lines of code. It had 15 data points. The other (DS3) contained projects consisting of 50 K or more lines of code. It had eight data points. One of the original projects included in DS3 was eliminated because it involved an unusually large amount of reused code. It was replaced by a large project which actually consisted of eight of the smaller



0730-3157/85/0000/0221$01.00 © 1985 IEEE

projects in DS1. The data used in each set may be found in [10].

Effort in this study is expressed in terms of staff-months. It consists of total programmer and management time for a given project. One staff-month of effort is defined as 160 staff-hours. Equations were derived with effort as the response variable. A list of the acronyms used is presented in the appendix. Table 1 shows the list of independent variables considered for regression. Definitions of terms used in the SEL may be found in [11] and the most important ones follow:

1. The total number of *lines of code* is the total number of lines of source code generated as a deliverable item for a project. It includes all executable, nonexecutable, and comment statements, whether newly coded or obtained from existing programs and library routines.
2. The number of *new lines* of code is the total number of lines of source code written by programmers for a given task. It excludes code taken from previously existing programs, but it includes comments, executable and non-executable statements.
3. The number of *modified lines* of code is the number of lines of previously developed code that has been changed for reuse in a new system.
4. The number of *developed lines* of code is defined in [8] as the number of new lines of code plus twenty percent of the number of reused lines of code. System integration and full system test are accounted for by the 20% overhead.
5. The total number of *modules* in a project is the number of independently compilable units such as FORTRAN functions, subroutines and BLOCK DATA, or separately identifiable and retrievable components from an on-line library.
6. The number of *new modules* is the number of modules that are not reused from some previous project.
7. The number of *modified modules* is the number of previously developed modules that has been modified in some way for reuse in a new system.
8. A *component* is a named piece of a system. Examples are a separately compilable function, a functional subsystem, or a shared section of data such as a COMMON block.
9. The number of *computer runs* is determined by the computer accounting systems and includes every job submittal for batch systems and every terminal sign-on for interactive systems.
10. The number of *pages of documentation* consists of written material, excluding source code statements and comments embedded therein, that describes a system or any of its components. It includes the program design document, development plan, test plans, system description, module descriptions and user's guide.

Newmodsq (*newmods*$^2$) and newratio (newlines / newmods) were variables suggested by Boydston [7]. They were found to have some significance on effort in studies made at the IBM Santa Teresa Laboratory.

The stepwise regression procedure of the Statistical Analysis System (SAS) package was initially used. More specifically, the maximum $R^2$ improvement (MAXR) technique was used for exploratory analysis. From the results, further analysis was done using the general linear models (GLM) procedure, and some plots were generated for single independent variable models that would possibly be useful in predicting effort during the early stages of development or models that showed a strong correlation between effort and the independent variable whose value cannot be determined early in the development. The latter cannot be used for resource estimation purposes but may be useful in characterizing effort in the environment.

The maximum $R^2$ improvement technique is considered almost as good as all possible regressions [12]. It tries to find the best one-variable model, the best two-variable model, and so on. Initially, it finds the one-variable model that yields the highest value for $R^2$. Another variable which gives the greatest increase in $R^2$ is added. After obtaining the two-variable model, each of the variables in the model is compared to each variable that is not in the model. For each comparison, the technique decides if deleting a variable and replacing it with the other variable results in an increase in $R^2$. When all possible switches have been compared, the one producing the maximum increase in $R^2$ is made. At this point, comparisons are made again. This continues until the technique can no longer find any switch that could increase $R^2$. Therefore the two-variable model generated is considered the best that the maximum $R^2$ improvement technique can find. One more variable is then added to the model, and the comparing-and-switching process is repeated until the best three-variable model is found. When there are no more variables that can be added to the model to increase the value of $R^2$, the procedure stops.

Table 1 - Variables Considered for Determining Effort

| |
|---|
| number of developed lines of code |
| number of pages of documentation |
| number of modified lines of code |
| number of modified modules |
| number of new lines of code |
| number of new modules |
| ratio of new lines of code to new modules |
| number of source code changes (versions) |
| number of components |
| number of computer runs |
| total number of lines of code |
| total number of modules |

## 3. ANALYSES AND RESULTS

Each data set was given several sets of candidate independent variables to be used in generating a model. The response variable used was effort.

For each set of candidate independent variables, the following steps were taken.

1. Run STEPWISE/MAXR to generate the best n-variable model, n = 1,2,...
2. Leave out of further consideration those models

with significance probability (Prob>F) > 0.05. Only consider those models where (Prob>F) <= 0.05 for the entire model and for each of the independent variables included in the model.

3. Disregard models where the ratio of the number of data points to the number of independent variables is less than 5 for DS1 and DS2. For DS3, because of the limited number of data points, consider models with up to 3 independent variables.

4. Disregard models with $R^2 < 0.5$. Preferably, $R^2$ should be >= 0.7 so that the model accounts for at least 70% of variation of effort in the model.

5. Disregard models with n variables where the increment of $R^2$ over that of the model with n-1 variables is very small. Do this for i = 2,...,n.

Across sets of candidate independent variables

1. Avoid models with higher-ordered terms.

2. Select model from the set with the greatest number of candidate independent variables originally supplied.

3. Select the model with the highest value of $R^2$ for the smallest number of variables.

The general linear models (GLM) procedure of SAS was used to examine in closer detail some of the more interesting one-variable models for each data set. Overlaid plots of predicted and actual values of effort versus the independent variables were generated. Plots of the residuals versus the independent variables were also produced.

### 3.1. All Projects

The first data set (DS1) consists of 23 projects ranging in size from 2.1 KLOC to 111.9 KLOC. The mean is 33.3 KLOC and the standard deviation is 32.5 KLOC. The number of modules ranges from 23 to 535 with a mean value of 198 and a standard deviation of 172. Effort for these projects ranges from 2.4 to 121.7 staff months. Mean effort is 40.9 staff-months and the standard deviation is 40.4 staff-months. Because of the wide ranges and the large standard deviations, the two smaller data sets (DS2 and DS3) were subsequently formed and analyzed separately.

Of all the sets of candidate independent variables used to generate a model of effort for this data set, only those which gave reasonable and interesting results are presented here. Initially, the set of candidate independent variables consisted only of newlines, newmods, modlines, and modmods. These are analogous to variables used by Boydston [7]. They can be determined in the early stages of project development and may therefore have predictive value. The one-variable equation that resulted is

$$Effort = 5.497 + 1.500\ newlines \qquad (1)$$

$$R^2 = 0.795 \quad F = 81.65 \quad Prob > F = 0.0001$$

The standard error of estimate (SEE) for the slope of this equation is 0.166. Adding newratio to the set of candidate independent variables yielded the same result. Figure 1 shows a plot of actual effort versus newlines for the different projects. The letters in the figure represent the different projects. Some observations are hidden due to overlap in values. The figure also shows the corresponding points predicted by equation (1). These are represented by asterisks (*) in the plot. The plot of the residuals versus newlines, which may be found in [10], does not show any systematic pattern of deviation. This suggests that the linear model may be adequate for describing the relationship. (Residual plots for other models developed in this study may similarly be found in [10].)

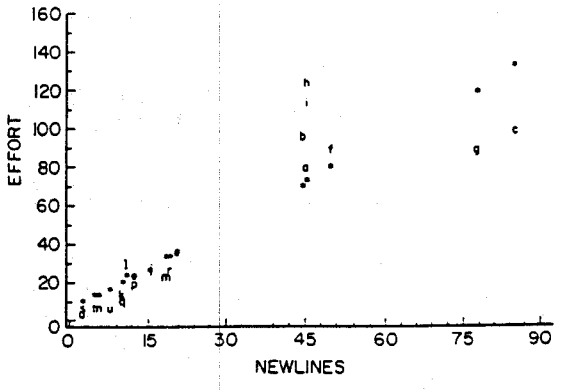
This plot shows a few points for which there is a large discrepancy between the actual and the predicted values of effort. Projects *h* and *i* for which the equation underpredicts the value of effort were both developed when there was a major change in the environment. A more reliable machine and more computer terminals were installed. There was quicker turnaround. The staff were turned loose on the computer. However, the level of experience of the staff for both these projects was lower than for most of the other projects in the study. Project *h* was a problem project. It did not have enough experienced staff to begin with and staffing adjustments had to be made in midstream. It was very late compared to other projects and was undertested. Project *i* was also a potential problem project, but was given more attention because of the unhappy experience with project *h* and was fortunately straightened out sooner. Projects *c* and *g* for which the equation (1) overpredicts effort were both developed with more experienced staff.

Because the number of developed lines was originally found by Bailey and Basili [8] to be the best predictor of effort in their meta-model for the SEL, it was added to the set of candidate independent variables. The resulting equation is

$$Effort = 4.372 + 1.430\ devlines \qquad (2)$$

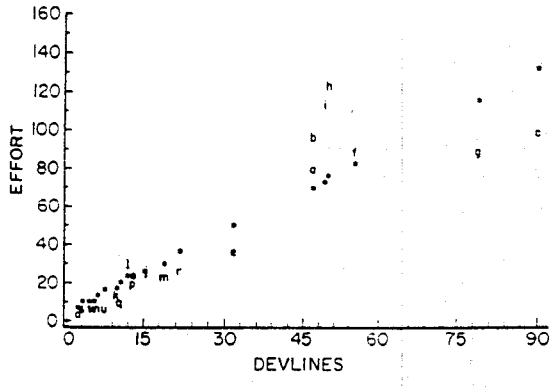
$$R^2 = 0.808 \quad F = 88.30 \quad Prob > F = 0.0001$$

The SEE for the slope of this equation is 0.152. Figure 2 shows the plot of actual and predicted values of effort versus




Notes:
1. Actual effort vs. newlines---symbol used is letter code of project.
2. Predicted effort vs. newlines---symbol used is *.
3. 9 observations hidden.

Figure 1 - Effort vs. newlines for DS1.




Notes:
1. Actual effort vs. devlines---symbol used is letter code of project.
2. Predicted effort vs. devlines---symbol used is *.
3. 4 observations hidden.

Figure 2 - Effort vs. devlines for DS1.

developed lines for equation (2). The outlier points are the same as those for equation (1). There is little difference in these two models and they indicate that the number of developed lines is at least as good as the number of new lines for predicting effort in the SEL.

Another set of models was generated using newlines, newmods, modlines, modmods and the squares of each of these. Boydston [7] found that there is a quantified square root trade-off between the number of new modules and the amount of new code per module and thus included a $newmods^2$ term in one of his effort equations. An attempt was made to determine whether or not the inclusion of the squared terms would have any effect on the effort model. The one-variable model that resulted is the same as (1) above. Two and three variable models were also generated as follows:

$$Effort = -11.938 + 3.427\ newlines - 0.025\ newlinsq \quad (3)$$

$$R^2 = 0.916 \quad F = 109.70 \quad Prob > F = 0.0001$$

$$Effort = -13.740 + 3.258\ newlines + 0.355\ modmods - 0.028\ newlinsq \quad (4)$$

$$R^2 = 0.933 \quad F = 88.53 \quad Prob > F = 0.0001$$

There is a substantial increase in $R^2$ in going from equation (1) to (3), but not from (3) to (4). This suggests that the quadratic equation (3) may be a much better model than (1), but the inclusion of the additional term modmods does not improve the model tremendously and only adds to the complexity.

Similarly adding the square of developed lines to equation (2) to parallel equation (3) yields the following model:

$$Effort = -10.588 + 2.992\ devlines - 0.020\ devlinsq \quad (5)$$

$$R^2 = 0.900 \quad F = 89.81 \quad Prob > F = 0.0001$$

This result shows a considerable improvement in $R^2$ over equation (2). Comparing equations (3) and (5) again shows little difference in the predictive power of new lines and developed lines.

In this study, in addition to predictive models of effort, relationships between effort and other variables in the SEL database were also sought. Models for characterizing and evaluating effort could enhance our understanding of the software development process in this environment. A set of models using as candidate independent variables all the variables in Table 1 with the exception of newratio was generated. Newratio was already found to be insignificant so it was excluded. Models with up to three variables were reasonable in this case and are presented here.

$$Effort = 9.951 + 0.008\ numruns \quad (6)$$

$$R^2 = 0.895 \quad F = 179.30 \quad Prob > F = 0.0001$$

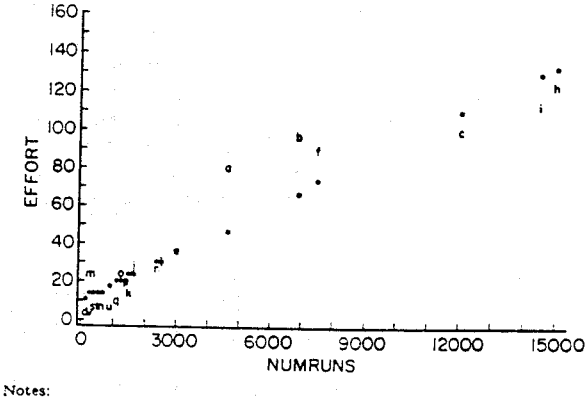
$$Effort = 3.384 + 0.104\ newmods + 0.006\ numruns \quad (7)$$

$$R^2 = 0.939 \quad F = 154.57 \quad Prob > F = 0.0001$$

$$Effort = 4.484 - 0.637\ modmods + 0.963\ devlines + 0.007\ numruns \quad (8)$$

$$R^2 = 0.978 \quad F = 285.34 \quad Prob > F = 0.0001$$

The one-variable model shows a very strong relationship between the number of runs in the project and the amount of effort. Numruns, by itself, accounts for 94% of the variation in effort. The SEE for the slope of equation (6) is 0.0006. Figure 3 shows the plot of actual and predicted effort versus number of runs for equation (6). For project *a* there is more actual effort per number of runs. It is one of the earliest projects included in this study. The developers




Notes:
1. Actual effort vs. numruns---symbol used is letter code of project.
2. Predicted effort vs. numruns---symbol used is *.
3. 5 observations hidden.

Figure 3 - Effort vs. numruns for DS1.

were relatively inexperienced. This project was also characterized by staffing and management problems early in the project and serious staffing changes. On the other hand, project *b* was developed by experienced staff. In this case, the actual effort is also higher than that predicted by the equation, probably because the developers were more thorough and purposely put in more effort per run.

To see what other variables correlate well with effort, the number of runs was excluded from the set of candidate independent variables. The following one-variable and four-variable models were generated:

$$Effort = 0.581 + 0.045\ docpages \tag{9}$$

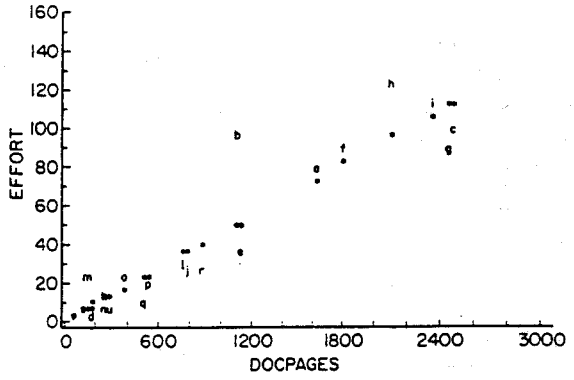
$$R^2 = 0.871 \quad F = 141.82 \quad Prob > F = 0.0001$$

$$Effort = 2.634 - 0.346\ newmods - 5.076\ modlines + 0.045\ docpages + 0.066\ numchngs \tag{10}$$

$$R^2 = 0.930 \quad F = 59.99 \quad Prob > F = 0.0001$$

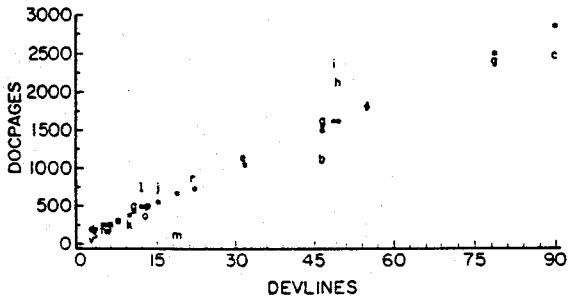
The one-variable equation shows the strong relationship that characterizes effort across these projects and pages of documentation. Like number of runs, however, this relationship cannot be used for predictive purposes since the number of pages of documentation cannot really be determined early in the project. The SEE for the slope of equation (9) is 0.0038. Figure 4 shows a plot of effort versus pages of documentation for equation (9).

The high correlation of both number of runs and pages of documentation with effort led to the investigation of whether or not a predictor variable like the number of developed lines could be used to predict their values. If the number of runs a project would entail could be determined ahead of time, this information could be used to allocate computer time. Similarly, if a good estimate of the number of pages of documentation could be obtained during the early stages of project development, the publications group could be made ready. The following result was obtained for




Notes:
1. Actual effort vs. docpages—symbol used is letter code of project.
2. Predicted effort vs. docpages—symbol used is *.
3. 3 observations hidden.

Figure 4 - Effort vs. docpages for DS1.




Notes:
1. Actual docpages vs. devlines—symbol used is letter code of project.
2. Predicted docpages vs. devlines—symbol used is *.
3. 2 observations hidden.

Figure 5 - Docpages vs. devlines for DS1.

the number of runs:

$$Numruns = -108.274 + 150.879\ devlines \tag{11}$$

$$R^2 = 0.686 \quad F = 45.90 \quad Prob > F = 0.0001$$

The SEE for the slope of equation (11) is 22.269. The number of developed lines does not explain more than 69% of the variation in number of runs. A much better result was obtained for pages of documentation:

$$Docpages = 99.143 + 30.895\ devlines \tag{12}$$

$$R^2 = 0.892 \quad F = 173.24 \quad Prob > F = 0.0001$$

The SEE for the slope of equation (12) is 2.347. Figure 5 shows a plot of actual and predicted pages of documentation versus number of developed lines based on equation (12). The number of developed lines accounts for almost 90% of the variation in pages of documentation. Basili and Freburger [6] obtained the following equation from a smaller set of projects in the SEL:

$$Doc = 34.7\ (DL^{0.93})$$

where

$$Doc = pages\ of\ documentation$$

$$DL = number\ of\ developed\ lines$$

They noted that the relationship is approximately linear and the above result tends to support this observation.

Deleting both numruns and docpages from the independent variable set yields two reasonable models for effort. The one-variable model is the same as equation (2) above. The four-variable model generated is

$$Effort = 5.433 - 1.082\ newmods + 22.376\ newlines + 0.854\ totmods - 20.476\ devlines \tag{13}$$

$$R^2 = 0.926 \quad F = 56.24 \quad Prob > F = 0.0001$$

Many other combinations of variables were used but they either failed to produce any interesting results or they yielded the same results as the case above which included all variables in Table 1 with the exception of newratio. The number of runs is the independent variable most highly correlated with effort when all 23 projects are considered. It may be an excellent measure of the complexity of the project, the quality of the development, the quality of the product, the amount of testing involved, the level of structure or disorganization of project management, and a variety of other factors.

### 3.2. Projects Under 50 K Lines of Code

There are 15 projects with less than 50 K total lines of code. They range in size from 2.1 to 32.8 KLOC with a mean of 11.9 KLOC and a standard deviation of 8.1 KLOC. There are from 23 to 263 modules. The mean number is 91 and the standard deviation is 63. Effort ranges from 2.4 to 29.0 staff-months with a mean of 14.6 and a standard deviation of 9.5.

In all cases where the number of new lines was included in the set of candidate independent variables, the following model was generated:

$$Effort = 0.877 + 1.535\ newlines \qquad (14)$$

$$R^2 = 0.802 \quad F = 52.71 \quad Prob > F = 0.0001$$

This is so even where the number of runs was included. This equation is selected by the STEPWISE/MAXR technique as the best single-variable effort equation for the smaller projects. It has predictive power since the number of new lines can be estimated early in the development of the project. The SEE for equation (14) is 0.211. The plot of effort versus new lines for this equation is shown in Figure 6.

Where newlines was not included in the set of candidate independent variables, the number of developed lines was selected by the technique. The equation generated is

$$Effort = 1.013 + 1.423\ devlines \qquad (15)$$

$$R^2 = 0.797 \quad F = 50.92 \quad Prob > F = 0.0001$$

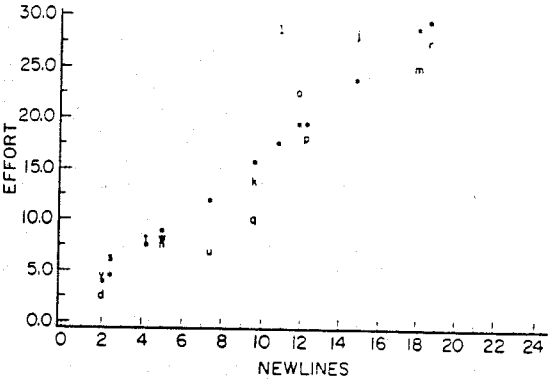
The SEE for the slope of this equation is 0.199. Figure 7 shows the plot of effort versus the number of developed lines. It is very similar to Figure 6. In the absence of new lines or developed lines, the number of total lines was selected as the predictor variable.

The two-variable equation generated in most attempted cases is

$$Effort = -1.185 + 0.108\ newmods + 0.009\ numruns \qquad (16)$$

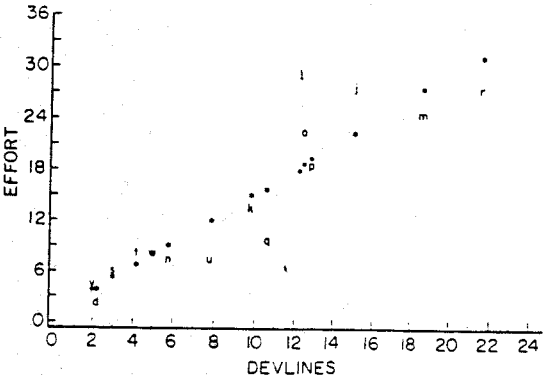
$$R^2 = 0.890 \quad F = 48.53 \quad Prob > F = 0.0001$$

The number of runs entered the model and the number of new modules replaced the number of new lines. None of the other models generated were reasonable.




Notes:
1. Actual effort vs. newlines---symbol used is letter code of project.
2. Predicted effort vs. newlines---symbol used is *.
3. 2 observations hidden.

Figure 6 - Effort vs. newlines for DS2.




Notes:
1. Actual effort vs. devlines---symbol used is letter code of project.
2. Predicted effort vs. devlines---symbol used is *.

Figure 7 - Effort vs. devlines for DS2.

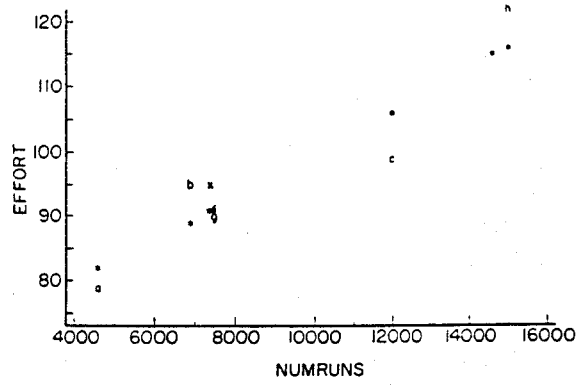
### 3.3. Projects with 50 K Lines of Code or More

There are 8 projects in this data set. The smallest consists of 50.9 KLOC and the largest is 111.9 KLOC. The mean size is 75.2 KLOC with a standard deviation of 19.9 KLOC. The number of modules ranges from 201 to 604 with a mean value of 427 and a standard deviation of 139. Effort ranges from 78.7 to 121.7 staff-months with a mean of 97.7 and a standard deviation of 13.5.

In all cases where the number of runs was included in the set of candidate independent variables, the following was generated as the best one-variable model:

$$Effort = 66.868 + 0.003\ numruns \qquad (17)$$

$$R^2 = 0.878 \quad F = 43.27 \quad Prob > F = 0.0006$$




Notes:
1. Actual effort vs. numruns---symbol used is letter code of project.
2. Predicted effort vs. numruns---symbol used is *.
3. 1 observation hidden.

Figure 8 - Effort vs. numruns for DS3.

The standard error of estimate for the slope of this equation is 0.0005. Figure 8 shows the plot of effort versus the number of runs for the larger projects.

All models which excluded number of runs from the set of independent variables yielded the following as the only reasonably good equation:

$$Effort = 122.220 + 1.088\ modmods - 0.960\ newlines - 3.883\ modlines \quad (18)$$

$$R^2 = 0.917 \quad F = 14.78 \quad Prob > F = 0.0125$$

For the larger projects, the number of modified modules is always the first independent variable selected for entry into the model. It seems to be a better predictor of effort for this environment than lines of code, whether new, developed or total, but not much better. It explains only 17% of the variation in effort. It seems that none of the variables which can be determined early in project development is a good single predictor of effort in larger projects.

There were no good two-variable models generated. Considering there are only 8 data points, caution should be exercised in using equation (18) for predictive purposes.

## 4. SUMMARY

As was shown in [8], developed lines of code is a good overall predictor of effort across all the projects considered in this study. It is one of the variables that can be estimated early in the project development and can thus be used to predict the effort requirements. For projects under 50 KLOC, the number of new lines was found to be the most significant predictor of effort whenever it was included in the set of candidate independent variables. The number of developed lines similarly predicts effort well for the smaller projects. The number of modified modules was not found to be most significant as a single predictor of effort in the Software Engineering Laboratory data. This differs from the result obtained by Boydston [7] in the IBM Santa Teresa Laboratory environment. Although the amount of code modification in the SEL was by no means small, it probably was not sufficient to show significance. For the projects that contained 50 KLOC or more, the number of modified modules is a better single predictor of effort than any of the line measures. It produced a better model of effort than newlines, devlines, modlines or totlines. However, it only accounts for 17% of the variation in effort and is clearly not by itself a good predictor of effort.

There is a high correlation between effort and number of runs overall and in the dataset containing projects that are at least 50 KLOC. There is also a high correlation between effort and pages of documentation across all the projects included in this study. Although these variables cannot be determined in the early stages of project development and therefore cannot be used for predictive purposes, they are nevertheless valuable for explaining and evaluating effort requirements in a project.

Only linear models, for the most part, were attempted for the sake of simplicity in this exploratory study. It would be premature at this point to select one of the above models as the best one to characterize, evaluate or predict effort. The selected model must be subjected to some test of stability. In this study, no suitable substitutes for lines of code in a baseline equation to predict effort were found. However, this study does give us some indication of other variables in the SEL database which are highly correlated with effort. From the non-predictive models generated, valuable insight into the software development process in the SEL environment was obtained.

## ACKNOWLEDGEMENTS

Research for this study was supported in part by the National Aeronautics and Space Administration under grant NSG-5123 to the University of Maryland. Computer support was provided in part by the Computer Science Center at the University of Maryland.

The authors are grateful to Mr. Frank McGarry of NASA/Goddard Space Flight Center and Dr. Jerry Page of Computer Sciences Corporation for their invaluable help in providing the data for this study. They also wish to thank the members of their research group at the University of Maryland and Dr. C. Frank Starmer of Duke University who have provided assistance of one form or another in the preparation of this paper.

## APPENDIX - LIST OF ACRONYMS

| Acronym | Description |
|---|---|
| devlines | newlines + 0.2 * (totlines - newlines) (KLOC) |
| devlinsq | $devlines^2$ |
| docpages | number of pages of documentation |
| modlines | number of modified lines of code (KLOC) |
| modlinsq | $modlines^2$ |
| modmods | number of modified modules |
| modmodsq | $modmods^2$ |
| newlines | number of new lines of code (KLOC) |
| newlinsq | $newlines^2$ |
| newmods | number of new modules |
| newmodsq | $newmods^2$ |
| newratio | newlines / newmods |
| numchngs | number of source code changes (versions) |
| numcomps | number of components |
| numruns | number of computer runs |
| projcode | project code |
| projname | project name |
| totlines | total number of lines of code (KLOC) |
| totmods | total number of modules |
| allhrs | proghrs + mgmthrs + servhrs |
| mgmthrs | management work time (tenths of an hour) |
| prmghrs | proghrs + mgmthrs |
| proghrs | programmer work time (tenths of an hour) |
| servhrs | services work time (tenths of an hour) |
| allefrt | allhrs / 1600 (staff-months) |
| effort | prmghrs / 1600 (staff-months) |

## REFERENCES


[1] Wolverton, R., "The Cost of Developing Large Scale Software", IEEE Transactions on Computers 23, No. 6, 1974.

[2] Putnam, L., "A General Empirical Solution to the Macro Software Sizing and Estimating Problem", IEEE Transactions on Software Engineering 4, No. 4, 1978.

[3] Boehm, Barry W., Software Engineering Economics, Prentice-Hall, Englewood Cliffs, New Jersey, 1981.

[4] Walston, C. E. and C.P. Felix, "A Method of Programming Measurement and Estimation", IBM Systems Journal 16, No. 1, 1977.

[5] Jeffery, D. R. and M. J. Lawrence, "An Inter-organizational Comparison of Programming Productivity", Department of Information Systems, University of New South Wales, 1979.

[6] Basili, V. R. and K. Freburger, "Programming Measurement and Estimation in the Software Engineering Laboratory", Journal of Systems and Software, Vol. 2, No. 1, 1981, pp. 47-57.

[7] Boydston, Robert E., "Programming Cost Estimate: Is It Reasonable?", Proceedings of the Seventh ICSE, IEEE Computer Society Press, March 26-29 1984, pp. 153-159.

[8] Bailey, John W. and Victor R. Basili, "A Meta-Model for Software Development Resource Expenditures", Proceedings, Fifth International Conference on Software Engineering, 1981, pp. 107-116.

[9] Boehm, Barry W., "Software Engineering Economics", IEEE Transactions on Software Engineering 10, No. 1, 1984.

[10] Basili, Victor R. and N. Monina Panlilio-Yap, "Finding Relationships Between Effort and Other Variables in the SEL", Technical Report TR-1520, Computer Science Department, University of Maryland, July 1985.

[11] Babst, T. A., F. E. McGarry, and M. G. Rohleder, "Glossary of Software Engineering Laboratory Terms", SEL-82-105, Software Engineering Laboratory Series, NASA/Goddard Space Flight Center, October 1983.

[12] SAS Institute Inc., SAS User's Guide: Statistics, 1982 Edition, Cary, NC: SAS Institute Inc., 1982.

[13] McGarry, F. E., G. Page, D. N. Card, et al., "An Approach to Software Cost Estimation", SEL-83-001, Software Engineering Laboratory Series, NASA/Goddard Space Flight Center, February 1984.