1-1-1999

Using Data Envelopment Analysis for Evaluating Alternative Software Development Process Configurations

Timothy R. Anderson
Portland State University, tim.anderson@pdx.edu

Peter K. Ghavami
Portland State University

Citation Details

Anderson, T.R.; Ghavami, P.K., "Using data envelopment analysis for evaluating alternative software development process configurations," Management of Engineering and Technology, 1999. Technology and Innovation Management. PICMET '99. Portland International Conference on , vol.1, no., pp.447, 1999

Using Data Envelopment Analysis for Evaluating Alternative Software Development Process Configurations

Timothy R. Anderson, Peter K. Ghavami
Engineering Management Program, Portland State University, 1900 SW 4th Avenue, Suite LL-50, Portland, OR 97201- USA

Abstract - The goal of achieving better software depends on improvements in software development processes. This paper provides a non-parametric, quantitative methodology for evaluating alternative software development process configurations as a tool for process improvement. The methodology uses Data Envelopment Analysis (DEA) for comparing the impact of varying degrees of software inspection on project schedule, cost and quality. Since every phase of the development process can be inspected at varying levels, a large number of process combinations become possible. Thus a manager's job to compare and select an efficient process configuration can become tedious. This is especially the case when several process configurations are being evaluated by comparing several productivity measures. Fortunately, Data Envelopment Analysis (DEA) can be applied as a method to reduce the long list of candidates for best practices to a selected few process options for further analysis. In this study, 257 different process options were compared against each other. The input and output values for each process option were estimated and compared using DEA. Of this quantity, 7 models were identified as candidates for best practices. An extension to this model incorporated managerial preferences in the form of weight restrictions applied to input/output variables.

I. Introduction

Despite advances in software development methodologies, many software development organizations still struggle with increased complexity in determining an appropriate process model that improves their quality, cost and schedule. The objective of this paper is to apply DEA as a quantitative methodology for comparing various process models and arriving at a short list of candidates for a best development process model.

A software development process is typically decomposed into smaller tasks or phases. Development process at the end of each phase can be measured and compared according to quality, schedule, development cost and other metrics. Based on historical data available from prior projects and by using software economic models, one can estimate the results of these tasks and predict the overall outcome of the project with adequate certainty [7][17][15].

Changing a development process without any evaluation of its impact on these metrics can be detrimental to an organization's productivity. It is shown that Simulation models can accurately mimic the organization's environment to predict result of process changes [21]. In this paper, the effects of varying inspection effort in four phases of software development are studied. To estimate the outcome of each process change, and as a data generation tool, a spreadsheet was developed that relied on COCOMO equations [8] and the software company's internal metrics. The data derived from this tool was used in the subsequent DEA analyses.

Data Envelopment Analysis (DEA) has received considerable attention since Charnes, Cooper and Rhodes [11] published their paper describing DEA using linear programming formulations. Since then, several application examples and extensions have been published [12][22][3]. For example, Banker and Kemerer [18] among others [23] have examined factors affecting software development and maintenance productivity using DEA. These studies compared the efficiency of multiple organizations against one another. In contrast, our study focuses on comparing the efficiency of multiple process models available to a single software development organization. This approach is similar to a paper published by Shafer and Bradford [24], in which they estimated the productivity of various machine configurations in a single factory and compared those estimates using DEA. Analogous to configuring manufacturing cells, we simulated several software development process combinations that can be applied to a single software project. After estimating the output of each process, we applied DEA for comparison.

The rest of this paper is organized as follows: Section I, offers an introduction and overview of literature in the subject area, Section II presents the methodology and tools used for this research, Section III provides the results and interpretation of those results, Section IV discusses future research topics.

A. Literature Review

In 1981, Barry Boehm published his seminal work on estimating software development costs and schedules by introducing the COCOMO model. Since that time, several other estimating approaches have been developed including COCOMO II, a derivative of Boehm's earlier work [8].

Some researchers have relied on the number of function points instead of KLOC (K Lines of Code) as a software metric [1]. However, for the purpose of our methodology and for comparing multiple variations of a single project, using KLOC is sufficient and appropriate.

Marc Kellner [16] offered a new approach to estimation by applying process modeling at the individual process activity level. Meanwhile, as software estimation and process

modeling research evolved during the 80's, a relatively new benchmarking technique called Data Envelopment Analysis emerged. Studies by Banker [4][5][18], and others including Paradi [20], Mahmood and Pettingell [19] introduced new applications for DEA as a benchmarking tool for comparing software development productivity among multiple organizations.

In the software development process area, Bhandari [6]. explored methods for improving software development process by analyzing defect data. Fenton pointed out that software process metrics used in estimating software costs and duration require a rigorous scientific measurement approach to be valid [13]. In 1996, Mahmood and Pettingel [19] investigated the productivity of 78 commercial software development projects. The results showed that DEA can be used successfully to identify the efficient and inefficient development groups.

While research on software process productivity has been extensive [16][21][17][18], the investigations so far have relied on reducing the process productivity measurements into a single metric such as life-cycle cost or utility. For example, Kellner uses "Utility" of a software process as a weighted measure of the project's three key attributes: schedule, defects and cost. Likewise, Raffo compares multiple process configurations by reducing the results of each process into a single variable, i.e. "Utility" by use of piecewise linear utility function. This method forces the evaluator to apply strict weights to software metrics and assumes a-priori knowledge of the weights.

However, DEA can support comparison of multiple process configurations on multiple measurements as described in this study. An advantage of DEA is that it allows comparison without applying strict pricing weights to each variable. Thus, DEA can narrow the number of process options to a few candidates for best practices. Although determining the "best" process can be controversial and subject to interpretation [3], our approach allows managers to focus on a short list of process options as candidates for most efficient process.

II. METHODOLOGY

Initial steps in this investigation required gathering historical information about the company's previous software projects (size, schedule, number of programmers, cost, defect rate at each phase of development, defect detection capability at each phase, etc.). This information provided the input and functional parameters to our software estimation template.
Since the company in our investigation used the waterfall development method, we assumed a waterfall process for this study. However, the same methodology and tools can be applied with minor modifications to spiral, evolutionary and other development models.

The development process at this particular company consisted of eight phases: Functional Specification (FS), High Level Design (HLD), Low Level Design (LLD), Code (CODE), Unit Test (UT), Functional Verification (FV), System Verification (SV), and Field deployment (FIELD). For this study, we examined the effects of adding inspection to the initial four phases of development. We limited the inspection effort to 4 levels, High, Medium, Low or None. As a result, 256 hypothetical process configurations (or process scenarios) were possible. Adding the Basecase scenario (the current development process used at this organization) a total of 257 process configurations are possible.

A. Data Generation

A Microsoft Excel template was developed to generate data, i.e. estimates for cost, duration, effort, risk and quality for each of the 257 process configurations. Where applicable COCOMO formulations were used, and for the rest we relied on the software company's development parameters for functional relationships. Thus, this spreadsheet and the results are unique to the company in our study. Users who wish to apply this methodology to their organization are encouraged to use an estimation tool of their choice. We estimated the outcome of each of the 257 process configurations by calculating 7 process measurements. These measurements were:

1. *Effort, (EFFORT)*: the number of person weeks required to complete the project. This variable is estimated based on project size and amount of inspection.
2. *Duration, (SCHED:,* the duration of project in weeks. It is calculated based on programmer productivity.
3. *Development Cost, (COST):* the development cost which is affected by EFFORT and the number of programmers.
4. *Maintenance cost, (MAINT):* this is a function of software defects remaining and reflects the cost of maintaining the software product after its is released.
5. *Risk exposure, (RISK):* is the relative measure of risk, determined proportional to project duration and defects remained in the software.
6. *Defects discovered, (DEF_DIS):* indicates the number of defects discovered.
7. *Defects corrected, (DEF_COR):* is the number of defects corrected.

Other variables such as programmer skill, complexity, project size, tools and CPU speed were not used in DEA analysis, since they were identical across all process configurations. The final results were reviewed by an expert to ensure that they are satisfactory.

A correlation analysis of the estimate values is shown in Table I. The highly negative correlation between RISK and DEF_DIS or DEF_COR are easily explained. RISK increases as quality deteriorates. High positive correlation between COST and SCHED or EFFORT is understandable since (all else being equal) an increase in EFFORT will increase COST and SCHED.

Several DEA models were evaluated using an algebraic modeling language called AMPL. In this paper we will only discuss the results of two DEA formulations, the CCR model and the CCR model with weight restrictions.

B. Productivity Analysis using DEA

DEA is a non-parametric method for relative efficiency of different decision making units (DMUs). In this study each process configuration, denoted by its 4-letter inspection designation is an independent DMU. Table II indicates the naming convention for each DMU corresponding to its unique process configuration. For example, HMML represents a DMU which uses High level of inspection at the Functional Specification phase, followed by Medium level inspections for High and Low level design phases, followed by Low level inspection in the Code phase.

1. CCR Model: Charnes, Cooper and Rhodes [11] provided a linear programming formulation to measure efficiency of multiple DMUs (Charnes, Cooper and Rhodes 1978). They extended the simple ratio of output over input analysis to estimate technical efficiency and production frontier based on multiple outputs and inputs. The result of this analysis is a frontier formed by efficient process configurations using constant returns-to-scale.

TABLE II
PROCESS CONFIGURATION TYPES

FS Phase	HLD Phase	LLD Phase	Code Phase	DMU
None	None	None	None	NNNN
Low	None	None	None	LNNN
. . .	. . .	. . .	. . .	. . .
High	Medium	Medium	Low	HMML
High	Medium	Medium	Medium	HMMM
. . .	. . .	. . .	. . .	. . .
High	High	High	High	HHHH
BaseCase	BaseCase	BaseCase	BaseCase	BASECAS

Our early DEA models used 5 inputs and 2 output variables. EFFORT, SCHED, COST, MAINT and RISK were input variables. Output variables were quality metrics: DEF_COR and DEF_DET. Later on, RISK was transformed so it could be used as an output variable. The new set of DEA models produced similar results, namely the same set of efficient and inefficient process configurations. The efficiency scores were identical regardless of using RISK as input variable or using its transformations as output variable.

TABLE I
INPUT-OUTPUT CORRELATION TABLE

Variables/ DEA method	Input-1 Effort	Input-2 Cost	Input-3 Sched.	Input-4 Maint. Cost	Input-5 Project Risk	Output-1 Defects Correct	Output-2 Defects Discov.
Effort	1.000						
Cost	0.915	1.000					
Sched.	0.997	0.902	1.000				
Maint Cost	-0.684	-0.547	-0.715	1.000			
Project Risk	-0.645	-0.515	-0.678	0.996	1.000		
Defects Correct	0.721	0.575	0.749	-0.993	-0.980	1.000	
Defects Discov.	0.652	0.523	0.686	-0.997	-1.000	0.981	1.000

The formulation for the primal, input-oriented CCR model is shown below: (1)

$$\begin{aligned} &Min\ \theta, \\ &\theta, \lambda \\ &s.t.\quad Y\lambda \geq Y_{\circ}, \\ &\qquad X\lambda \leq \theta X_{\circ}, \\ &\qquad\quad \lambda \geq 0; \end{aligned}$$

The dual model has the following formulation for the input-oriented model is given in (2): (2)

$$\begin{aligned} &Min\ Z = \sum \nu_i X_{i\circ}, \\ &\mu, \nu \\ &s.t.\quad \sum \nu_i X_{ij} \geq \sum \mu_r Y_{rj}, \quad j = 1,\ldots,n \\ &\qquad \sum \mu_r Y_{rj} = 1.0; \qquad r = 1,\ldots,s;\ i = 1,\ldots,m \\ &\qquad \mu, \nu \geq 0; \end{aligned}$$

2. DEA With Weight Restrictions: The CCR model allows all input and output variables to be equally important. However, in software development projects, time to market takes precedence over cost and often times over quality. These preferences are imposed by market conditions or environmental factors which can be outside of manager's control. Thus, incorporating these preference structures into DEA becomes necessary.

Paradi, Reese and Rosen [20] investigated several methods to determine appropriate weight restrictions. One method starts with considering inequality relationships among productivity ratios, such as DEF_DIS/EFFORT and DEF_COR/COST. From these relationships, Paradi, et. al. generate a system of linear equations that can be solved to determine weight restrictions. The other method, which is also used in this paper uses the economic or managerial emphasis towards each input/output variable.

Manager's preference structure can be applied via weights applied to input and output variables. Weight restrictions can be applied to the CCR model as lower bound and upper bound limits to reflect managerial pricing preferences. In the above formulation, μ and ν are weights applied to output and input variables respectively. To select weights appropriately, it is necessary to define a meaningful and realistic transformation for prices and costs of input and output variables.

User-specified weights, which reflect the preference over the adjustments of inputs or outputs require some a priori knowledge or assumptions about the variables. Selecting weights, i.e coming up with values for μ and ν can be a practical difficulty. One approach is to rank input or input variables according to their importance using the Analytic Hierarchy Process to obtain a set of ordinal weights. In this study we used both the economic relationship and hierarchy between variables to indicate the preference structure (Paradi 1997). The input variables, COST, MAINT, EFFORT and SCHED have economic trade-offs. The output variables, DEF_COR and DEF_DIS can be related via importance hierarchy.

For example, it is conceivable that a manager might be willing to pay more for programmers in order to improve the scheduled completion date, by outsourcing part of the project. In other words, the manager perceives SCHED to be more important than EFFORT. This preference structure reflects his/her willingness to pay the additional COST of EFFORT to reduce SCHED.

In situations where skilled contractors are scarce, consider a manager who is in a bind and is willing to pay 4 times (but less than 5 times) the regular programmer wages to improve her schedule by one week. This hypothetical situation puts strict upper and lower bounds on weights. In other words, $\mathbf{v_3} / \mathbf{v_2} \geq 4$, and $\mathbf{v_3} / \mathbf{v_2} \leq 5$. But in the case of this project that employs 7 developers, one week improvement equates to gaining 7 person weeks. The correct economic relation implies that gaining one week of SCHED is equivalent to hiring 7 programmers, i.e. the weight constraints should be formulated as $\mathbf{v_3} \geq \mathbf{7 * 4 * v_2}$, and $\mathbf{v_3} \leq \mathbf{7 * 5 * v_2}$. Of course, not all preference structures need to be as restrictive as in this scenario. In fact if the weight restrictions are too restrictive, the DEA model will result in no feasible solution.

Similarly, management might believe that correcting a defect is 2 to 3 times as resource intensive as finding a defect, hence the weight ratios of $\mu_1 \geq \mathbf{2} * \mu_2$ and $\mu_1 \leq \mathbf{3} * \mu_2$.

III. Results

As discussed earlier, the CCR model revealed 7 efficient process configurations. Contrary to our initial expectation, the base case which is the current software development practice at this company, was not among the efficient group. In fact the efficiency score for the base case (DMU257) was below the average score of .836.

A closer examination of efficiency scores provides interesting findings: For example, while MHHH (DMU160) was selected to be on the efficiency frontier, other combinations such as HHHM (DMU237) were not deemed as efficient. But even this DMU scored higher than the base case. The median of CCR efficiency scores came to .856, still higher than base case efficiency score. In fact, compared to the base case, at least 219 DMUs were more efficient than the base case, and only about a dozen process configurations performed worse than the base case.

From a software development perspective, it is more economical to detect and correct as many errors as early as possible. This preference justifies a heavier bias towards corrected defects over detected defects. Another preference

might be given to project duration (schedule) over effort. These preferences were handled by imposing weight constraints in the second CCR model.

A review of efficiency scores suggests that DEA can be successfully used for evaluating the overall performance of these process configurations. It can also provide paths to efficient DMUs by indicating the amount of increase or decrease in input/output variables. For example, consider process configuration NHHH (DMU64). This DMU is dominated by its target DMU, MMHH (DMU191). Compared to its target, this DMU must decrease its input by 5%, 15%, 2% and 4% respectively for EFFORT, COST, SCHED and RISK.

A selected list of process estimates and their efficiency scores, including all DMUs with efficiency score of 1.0 are shown in Table III.

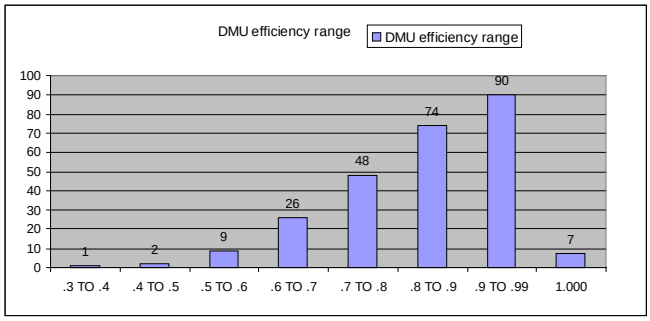
TABLE III
PROCESS CONFIGURATION INPUT-OUTPUT VALUES AND CORRESPONDING EFFICIENCY SCORES

DMU Config.	Seq. No.	EFFORT Weeks	COST ($000)	SCHED Weeks	MAINT Cost ($000)	RISK	Total DEF_COR	Total DEF_DIS	CCR Model No weight restrictions Efficiency	CCR Model Weight Restrictions Efficiency
NHHH	64	290.38	421.75	46.25	12	73	221	235	0.9857	0.9857
LHMH	116	269.48	424.43	45.05	14	78	219	233	0.9940	0.9940
LHHH	128	282.76	399.89	45.82	12	69	221	235	0.9934	0.9934
MLNN	139	188.17	297.29	39.73	162	707	89	107	0.6140	0.5892
MHMH	153	264.72	408.80	44.77	14	75	219	234	1.0000	1.0000
MHML	154	223.88	386.42	42.22	62	242	165	200	0.9474	0.9210
MHMM	155	232.25	396.59	42.77	38	149	189	219	1.0000	1.0000
MHMN	156	217.00	374.21	41.76	97	390	135	170	0.8308	0.7888
MHHM	157	251.82	384.03	44.00	31	120	196	225	0.9905	0.9623
MHHH	160	279.04	386.20	45.61	12	65	221	236	1.0000	1.0000
MNNN	161	192.58	296.09	40.06	178	791	79	90	0.5102	0.5023
MMNN	179	186.34	302.91	39.60	149	641	98	120	0.6909	0.6597
MMLL	181	217.39	360.39	41.79	80	325	151	183	0.8990	0.8679
MMLN	182	195.49	319.55	40.27	124	520	116	144	0.7893	0.7482
MMMH	185	270.81	401.16	45.13	15	81	218	233	0.9896	0.9888
MMMN	186	204.75	331.57	40.92	104	430	131	162	0.8496	0.8056
MMML	187	218.11	356.13	41.84	68	268	161	195	0.9582	0.9221
MMMM	188	231.87	376.60	42.74	41	165	187	215	0.9971	0.9899
MMHN	189	236.44	342.50	43.04	83	335	148	182	0.8471	0.7919
MMHL	190	240.09	348.56	43.27	54	209	174	207	0.9484	0.9064
MMHH	191	275.01	359.66	45.38	12	70	221	235	1.0000	1.0000
MMHM	192	245.77	353.60	43.62	33	131	195	223	1.0000	0.9798
HNLM	208	286.93	464.27	46.05	55	228	176	204	0.8476	0.8044
HMNH	211	309.37	471.85	47.28	20	104	214	229	0.9267	0.9258
HMMM	217	243.61	389.57	43.49	40	155	188	218	0.9742	0.9540
HMMN	218	220.86	352.87	42.02	100	407	132	167	0.8145	0.7592
HMML	219	231.84	372.89	42.74	65	252	162	198	0.9185	0.8777
HMHM	221	259.60	370.42	44.47	32	123	195	224	0.9756	0.9363
HMHH	224	288.10	375.05	46.12	12	67	221	236	1.0000	0.9940
HHHM	237	270.04	406.91	45.09	30	113	196	226	0.9597	0.9597
HHHH	240	297.01	408.62	46.61	11	63	221	237	1.0000	1.0000
HLHH	256	299.37	390.21	46.74	12	71	221	235	0.9779	0.9779
BaseC	257	252.46	402.90	44.04	114	484	125	152	0.6656	0.6246
Min Score		186.34	296.09	39.60	11.01	62.93	67.33	67.35	0.3985	0.3985
Max Score		473.69	775.97	54.89	199.22	904.08	221.03	236.74	1.0000	1.0000
Avg Score		264.80	419.68	44.64	66.74	286.65	168.01	191.60	0.8360	0.8127
No. of Efficient DMUs									7	5

A. Results Interpretation

The CCR model with weight restrictions, imposed price ratio limits on SCHED, DEF_COR and DEF_DIS. The CCR model selected only 7 process models (less than 3% of all process configurations) as efficient. When manager's preference in form of weight restrictions were applied, the list was reduced to 5 DMUs (less than 2% of all process configurations).

The efficiency scores have a distribution range from .399 to 1.000 with the majority of scores falling between .800 and .999 (Fig 1). The average efficiency score decreased from



.836 in CCR model to .813 when preference weights were applied.

Fig 1. Distribution of DMU efficiency scores

An important finding for the project manager was that the current practice, i.e. the BASECASE scenario was relatively inefficient under both DEA models. To illustrate this point, compare the BASECASE efficiency score in each DEA analysis with the mean efficiency scores. The BASECASE efficiency score is consistently below the average score.

We learned that the company in our study typically applies medium level inspection early in the development process and then ad-hoc levels of inspections (below the Low level) through-out the rest of the project. This practice is close to the MLLN (DMU130) configuration and understandably influenced by the conventional thinking that applying inspection to the functional specification phase is a key determinant of the project success. While this is true, it is insufficient for achieving highest productivity in software development.

The DEA results indicate that for this project, there are many more efficient practices possible. The company can gain better results if it raised its inspection efforts to at least a Medium level across all phases of development. For example, MMMM process configuration (DMU188) delivers almost 50% better output (less defects) than the current practice by using by 9%, 7%, 30%, and 64% less inputs of EFFORT, COST, SCHED, MAINT respectively. The DMUs which produced efficiency scores of 1.000 included at least one iteration of High level of inspection to any one of the development phases. To illustrate this, consider software development practices such as MMHM (DMU192), MHMM (DMU155), and MHMH (DMU153) which are far more efficient than the base case.

Every DMU is compared against a virtual DMU that is a convex combination of other DMUs. This convex combination is described by the λ values as in (1). The vast majority of inefficient DMUs were compared against three efficient DMUs: MMHM (DMU192), MHMM (DMU155), and MHMH (DMU153). This was evident from the λ values of inefficient DMUs, because they consisted of fractions of λ values from these three efficient DMUs. The results indicate that applying intense level of inspection at either the High Level Design, Low Level Design or the CODE phase achieves most efficient results. The MHMH process in particular dominated more than 75% of the inefficient DMUs.

In other models, additional weight restrictions can be imposed to reflect manager's preference for SCHED or COST relative to RISK.

B. Correlation of DEA results

Table IV shows results of correlation analysis between data variables and efficiency scores follow. A high negative correlation between efficiency scores and RISK or MAINT on one hand and high positive correlation with DEF_COR or DEF_DIS are the indication that these values played a more significant role in DEA's evaluation of DMUs.

TABLE IV
CORRELATION OF DEA RESULTS AND INPUT-OUTPUT VARIABLES

Variables/ DEA method	Input-1 Effort	Input-2 Cost	Input-3 Sched.	Input-4 Maint.	Input-5 Project Risk	Output-1 Defects DEF_COR	Output-2 Defects DEF_DIS
CCR model, Input-Oriented/ No weight	0.349279	0.232376	0.383861	-0.91242	-0.93394	0.876313	0.930362
Dual DEA Input model Weight constraints: S>=28C, DC>=2DD	0.388799	0.273266	0.419519	-0.92636	-0.93744	0.90437	0.934328

It is plausible that by introducing weight restrictions in favor of DEF_COR over DEF_DIS, that we find higher correlation between efficiency scores and DEF_COR. Indeed, the scores from weight restriction DEA model, indicate higher correlation with DEF_COR.

High correlation among the output variables is due to the design of this experiment, i.e. due to limited variation produced by the data generation model, because we compared the process configurations on a single project. Had we compared results of process changes amongst several companies, then the output variation would have been greater, and thus we could expect lower correlation between input-output values. Fortunately, DEA is robust enough to be immune to these high correlation values.

A broader comparison of the DEA results can be accomplished by comparing the range of values, i.e. the difference between the lowest and highest input-output values for efficient DMUs. This comparison is shown in Table V.

The normalized Relative Difference (RDIFF) between each variable for all efficient DMUs were calculated by (3), where $X_{o\ -max}$ and $X_{o\ -min}$ represent the high and low values for a given input-output variable of DMUs present on the efficiency frontier.

$$RDIFF = \sum \left| \frac{X_{\circ \max} - X_{\circ \min}}{X_{\max} - X_{\min}} \right| \quad (3)$$

As shown in Table V, the CCR model with weight restrictions has narrowed its selection of DMUs by giving more importance to SCHED, while allowing the COST to increase. This technique allows managers to interpret DEA's selection process. Note that the range of input-ou variables decreases as we use the DEA with we restrictions, i.e. DEA becomes more selective by narrow... its range of input-output values.

For example, the CCR model produced 7 efficient DMUs. The values for COST, an input variable varied between 353.6 and 408.62. In the weight restriction model, the values for COST varied between 359.66 and 408.62.

C. Summary of Results

DEA can be an excellent tool for reducing a large number of process configurations to a small set of candidates for best practices. One can realize the practical difficulties of actually coming up with a short list of efficient process configurations from a long list of 257 possibilities. The distribution of efficiency scores (Fig 1) underscores this difficulty, as 63% of DMUs have efficiency scores that fall some where between .8 to .99, but without DEA analysis appear as viable processes.

DEA with weight restrictions can be highly desirable and seems to have considerable potential for software project planners or process consultants, because A) the model further reduces the number of candidates for best practices, and B) It allows managers to explore various business policies by imposing various managerial preferences on either inputs or outputs. We have shown that by applying preference in form of weight restrictions, the list of candidate DMUs can be further reduced.

When DEA is applied to software development process configurations as discussed in this study, managers can arm themselves with the necessary evidence to make a case to upper management for changing their current processes. The results of the study can point managers to fewer selected process options to choose from.

TABLE V
INCREASE IN EFFICIENCY SCORES RELATIVE TO REDUCTION OF INPUT-OUTPUT RANGES

DEA Model	Input EFFORT	Input COST	Input SCHED	Input MAINT	Input RISK	Output DEF_COR	Output DEF_DET
CCR Model results							
MIN	232.25	353.60	42.77	11	63	189	219
MAX	297.01	408.62	46.61	38	149	221	237
RDIFF	22.5%	11.5%	25.1%	14.4%	10.2%	20.6%	10.6%
Above Mean	4	0	5	0	0	7	7
Below Mean	3	7	2	7	7	0	0
CCR with Weight Restrictions							
MIN	232.25	359.66	42.77	11	63	189	219
MAX	297.01	408.62	46.61	38	149	221	237
RDIFF	22.5%	10.2%	25.1%	14.4%	10.2%	20.6%	10.6%
Above Mean	3	0	4	0	0	5	5
Below Mean	2	5	1	5	5	0	0

IV. Future Research Topics

Software development is characterized as a highly non-linear activity with possibly variable return to scale characteristics. The CCR model is a constant return to scale model. Thus, for comparing efficiency of software development process configurations, the CCR formulation may not be an accurate model. For example, increasing the number of programmers does not necessary increase output proportionally. Instead, adding programmers to a project may produce diminishing effects on productivity, although not proportionally. Also, removing defects can cause the code to become fragile and eventually introduce other defects and "side-effects".

Other exciting applications of DEA in software development are possible. DEA might be used to measure productivity of individual programmers on a specific project, or efficiency of structured programming vs. object oriented programming technique over several development projects. Another extension to this study may include the application of Monte Carlo simulation as a technique to generate better data for DEA analysis.

References

[1] A. Abran, "Function Points Analysis: An Empirical Study of Its Measurement Processes," *IEEE Transactions on Software Engineering* 22(12, December 1996).

[2] P. Andersen and N. C. Petersen, "A procedure for ranking efficient units in data envelopment analysis," *Management Science* 39(10): 1261-4.

[3] T. R. Anderson and A. Uslu, "Selecting the "best" using data envelopment analysis," *Proceedings of PICMET '97*, Portland, PICMET.

[4] R. Banker and C. Kemerer, "Factors affecting software maintenance productivity: an exploratory study," *Proceedings of the 8th International Conference on Information Systems*, Pittsburgh, PA.

[5] R. D. Banker and C. F. Kemerer, "Scale economies in new software development," *IEEE Transactions on Software Engineering*, 15(10): 1199-205.

[6] I. Bhandari, "A Case Study of Softwae P{rocess Improvement During Development," *IEEE Transactions on Software Engineering* 19(12, December 1993).

[7] B. W. Boehm, *Software Engineering Economics*, Prentice Hall, 1981, page 75.

[8] B. W. Boehm, *Software Engineering Economics*, Prentice Hall, 1981, page 156.

[9] B. W. Boehm, "COCOMO II Model Definition Manual." *University of Southern California*, Project led by Barry Boehm.

[10] A. Charnes and W. W. Cooper , "The non-archimedean CCR ratio for efficiency analysis: a rejoinder to Boyd and Fare." *European Journal of Operational Research* 15(3): 333-334.

[11] A. Charnes, W. W. Cooper, et al., "Measuring the efficiency of decision making units." *European Journal of Operations Research* 2(6): pp. 429-44.

[12] A. Charnes, W. W. Cooper, et al, "Classifying and characterizing efficiencies and inefficiencies in data development analysis." *Oper. Res. Lett. (Netherlands)* 5(3): 105-10.

[13] Fenton ,"Software Measurement: A Necessary Scientific Basis." *IEEE Transactions on Software Engineering* 30(3, March 1994).

[14] B. Golany and S. Thore, "Restricted best practice selection in DEA: An overview with a case study evaluating the socio-economic performance of nations", *Annals of Operations Research* 73(1997) 117-140.

[15] W. Harrison, "Software Engineering Process & Quality", *Portland State University Lecture Notes*. 1994

[16] M. Kellner, "Software Process Modeling: Value and experience." *SEI Technical Review, Software Engineering Institute, Carnegie Mellon University*.

[17] M. I. Kellner, M. D. Kasunic and M. S. Krishnana, "Process Improvement: An Analysis of Users' Needs." *Submitted position paper for ISPW-8, October 1992*.

[18] C. F. Kemerer, "Production process modeling of software maintenance productivity," *Proceedings of the Conference on Software Maintenance* - 1988 (IEEE Cat. No.88CH2615-3) , Scottsdale, AZ, USA 24-27 Oct. 1988.

[19] M. A. Mahmood, K. J. Pettingell, et al., "Measuring productivity of software projects: A data envelopment analysis approach," *Decision Sciences* 27(1): 57-80.

[20] J. C. Paradi, D. N. Reese and D. Rosen, "Applications of DEA to measure the efficiency of software production at two large Canadian banks", *Annals of Operations Research*, 73(1997) 91-115

[21] D. M. Raffo, "Modeling Software Processes Quantitatively And Assessing the Impact of Potential Process Changes on Process Performance," *PhD dissertation*, Carnegie Mellon University, 1996

[22] L. M. Seiford and R. M. Thrall, "Recent developments in DEA: the mathematical programming appoach to frontier analysis." *Journal of Econometrics* 46: 7-38.

[23] J. K. Sengupta, "Data envelopment analysis for efficiency measurement in the stochastic case," *Comput. & Oper. Res. (GB)* 14(2): 117-29.

[24] S. M. Shafer, and J. W. Bradford, "Efficiency measurement of alternative machine component grouping solutions via data envelopment analysis," *IEEE Transactions on Engineering Management* 42(2): 159-165.

[25] H. Tulkens, "On FDH efficiency analysis: some methodological issues and applications to retail banking, courts, and urban transit." *Journal of Productivity Analysis* 4: 183-210.

[26] H. Tulkens, and P. V. Eeckaut, "Non-parametric efficiency, progress and regress measures for panel data: Methodological aspects," *European Journal of Operational Research* 80(3): 474-499.