DOI: 10.1145/1435417.1435449

BY PARAG C. PENDHARKAR AND JAMES A. RODGER

The Relationship between Software Development Team Size and Software Development Cost

MOST OF THE SOFTWARE DEVELOPMENT COST is related to the programmers' salaries and recent evidence shows that team-related factors can affect project performance and its cost.[10, 12] Selecting an appropriate team size for software development projects continues to be a challenge for most project managers. Larger teams represent better distribution of skills, but lead to higher communication and coordination costs.[8] Smaller teams lead to lower communication and coordination costs, but lead to programming biases that make software maintenance costly. Smaller teams, due to homogeneous programmer backgrounds, may lack the problem solving expertise for large size complex projects. In addition to programmer skills, Gorla and Lam10 have noted that heterogeneity in team member personality may be desirable in certain phases of the SDLC. Additionally, Gorla and Lam[10] emphasize that diverse expertise in IT and business application experience is always desirable in any project.

Software development project tools provide advance coordination and communication capabilities and are often used to control communication and coordination costs for large sized teams. Without the use of software development tools, it is easy to show that team communication requirements increase exponentially with the increase in team size. Since most projects use advance software development tools, the assumption of exponential increase in software development cost with increase in team size will seldom hold true in real world. Even a linear relationship between software development cost and team size may be considered a mute point, at best.

Understanding the relationship and impact of team size on software development cost is important because most managers, when confronted with strict deadlines, are tempted to increase team size. Among the reasons for increasing team size are:[9, 6]

1. Pressure to meet the schedule, as competitive advantage of the technology decreases with time,
2. Assigning one person to a project does not mean linear completion time, and
3. Dividing work into small sub-projects allows many people to work on the project simultaneously, which reduces the project completion time.

Although increasing team size for strict deadlines appears to be a good strategy, it is not really clear if it works well. For example, Biffl and Gutjahr[5] mention that increasing team size will more likely increase the number of defects found by the team and may not necessarily mean that project will be completed on time. In fact, most software quality professionals admit that team size should be larger in the beginning or middle of the project than at the end of the project. The higher team sizes in the beginning of the project allow for strict testing of the software product in early phases of SDLC.

In this article, we study the relationship between team size and software development cost. Since larger size projects are likely to contain large team size and vice versa, we use software project size in function points (FP) as a contextual variable. We use real-world data from 540 software projects for our data analysis. Our data was obtained from the International Software Benchmarking Standards Group (ISBSG) and is publicly available.

Figure 1: ISBSG release 7 project data origin by country.

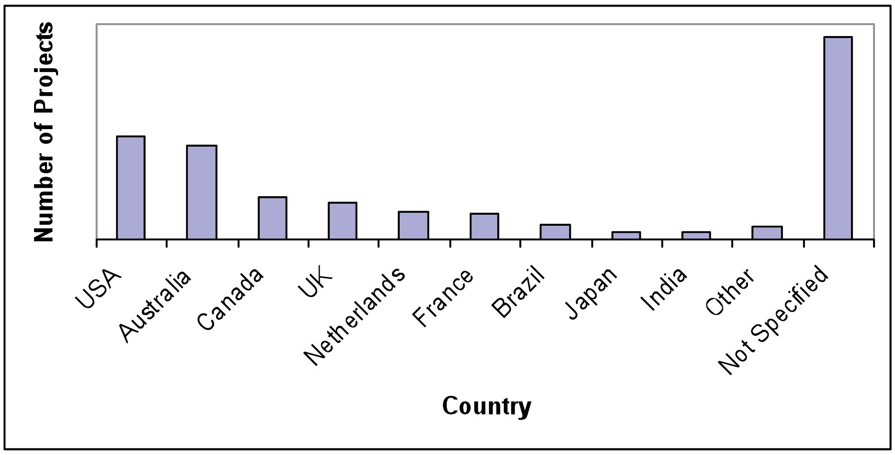
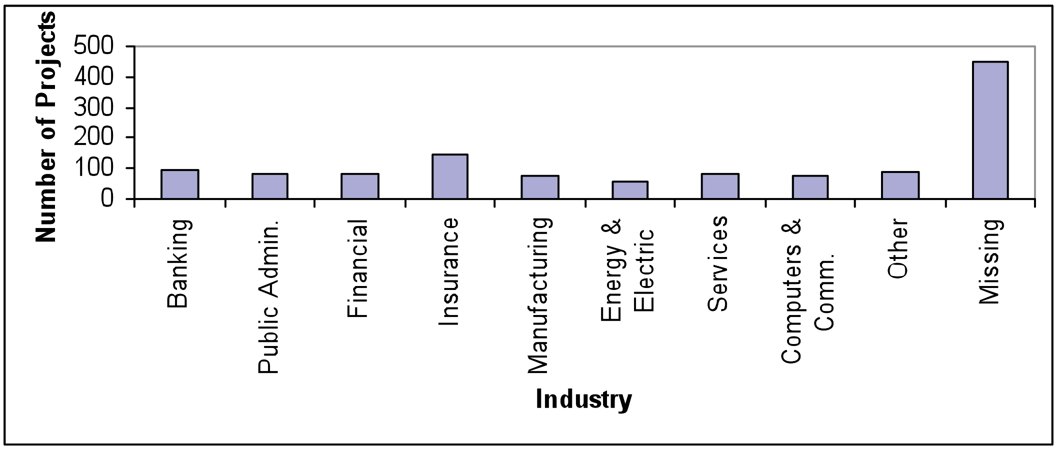


Figure 2: Industry type distribution in the ISBSG release 7 project data.



Data And Experiments

The ISBSG (release 7) data is used by several companies for benchmarking software projects and is available for sale to interested parties. The ISBSG procedures encourage software development teams to submit their project data to the repository in return for a free report, which graphically benchmarks their projects against similarly profiled projects in the ISBSG repository.[1] The software project data is submitted by the software project manager, who completes a series of special ISBSG data validation forms to report the confidence he/she has in the information he/she provides. ISBSG has developed a special mutually exclusive data quality rating that reflects the quality of data related to any given project. Each project is assigned a data quality rating of A, B, and C to denote the following:

- A= The project data satisfies all the criteria for seemingly sound data.
- B= The project data appears fundamentally sound, but some data attributes might not be fundamentally sound.
- C= The project data has some fundamental shortcomings.

The software projects in ISBSG release 7 data came from 20 different countries. Figure 1 illustrates the major data-contributing countries. The top three known contributing countries were the U.S., Australia, and Canada. Over 97% of the projects were completed between the years 1989-2001. Most of the projects (about 50%) were completed between the years 1999-2001. Figure 2 illustrates the industry type distribution for the 1,238 projects. Table 1 details the descriptive statistics of a few variables for the original 1,238 projects data.

Of the total 1,238 software projects, we first sorted the projects by data quality rating and deleted all the projects containing the data quality rating of C because the ISBSG recommends that data quality rating of A and B projects should be used for reliable data analysis. For the remaining projects, we selected all the projects that did not contain any missing value or zero value for either of three variables, which were team size, software size and software effort. After our data screening we had a total of 540 projects. Table 2 illustrates the descriptive statistics of the three variables for the 540 projects used in our data analysis. Comparing the mean, standard deviation, minimum and maximum values for FP in Tables 1 and 2 indicates that the 540 projects used in our analysis did not contain very large and very small size projects from the original 1,238 projects. Further, all the three variables in our sample, when compared to the original set of 1238 projects, had lower variance.

We use production economics theory as a guideline for testing the relationship between team size and software cost. We use software effort as a surrogate measure for software cost because once the software effort is known software cost can be computed by multiplying software effort with average hourly programmer wage. The existing literature in production function analysis

Table 1: Descriptive statistics of variables in ISBSG release 7 project data

Variable	Mean	Std. Dev.	Minimum	Maximum
Function Points	642.59 counts	1264.60	8	19050
Effort	8414.80 man-hours	32698.70	10	645694
Team Size	8.16 persons	20.16	1	468

Table 2: Descriptive statistics of variables.

Variable	Mean	Std. Dev.	Minimum	Maximum
Function Points	521.04 counts	846.46	11	9803
Effort	5271.70 man-hours	10873.93	17	150040
Team Size	8.13 persons	9.052	1	77

Table 3: Normality test results for regular and log-transformed variable values

			Kolmogorov-Smirnov	
Variable	Skewness	Kurtosis	Statistic	Significance
FP	4.905	36.023	0.276	0.000
Ln(FP)	0.373	-0.155	0.050	0.002
Effort	6.977	72.083	0.316	0.000
Ln(Effort)	-0.072	0.467	0.029	0.200
Team Size	3.364	15.274	0.216	0.000
Ln(Team Size)	0.165	-0.109	0.065	0.000

provides three models that can allow us to test the relationship between team size and software effort. These three models are linear regression, log-linear models to test non-linear relationships and non-parametric data envelopment analysis (DEA) model. Linear and log-linear models assume that the variables used in the models, regular values for linear regression and log-transformed values for log-linear models, are normally distributed. Using the SPSS software package, we conducted a Kolmogorov-Smirnov (KS) normality test for three variables and their natural logarithm transformed values to check if the variables and their transformations were normally distributed. The KS tests the null hypothesis that there is no difference between the variable distribution and normal distribution. Since a perfectly normal distribution has the skewness and kurtosis values of both equal to zero, we report the results of KS test, skewness and kurtosis values for the variable distributions in Table 3.

The results of our normality tests indicate that except for our test on log-transformed software effort variable, all the other tests indicated that the variable distributions were non-parametric. Since most of our variables violated normal distribution assumptions for linear and log-linear models, we did not use these models for our analysis. We decided to choose the last remaining non-parametric DEA model for our analysis. The DEA model does not assume normal distribution.

Banker and Kemerer[3] proposed the use DEA approach for production function estimation. Banker and Kemerer3 argued that the DEA approach is superior to parametric approaches, as it does not impose a particular form on the production function and assumes a monotonically increasing and convex relationship between inputs and outputs. Monotonicity and convexity are standard economic production function assumptions.[3]

To test the relationship between team size and software development cost, we use the non-parametric DEA statistical testing approach described in Banker and Slaughter.[4] We calculated the inefficiencies, θ^C and θ^B, for each i^{th} software project using the Charnes, Cooper and Rhodes (CCR) and Banker, Charnes and Cooper (BCC) models[7, 2] respectively. We assume that the reader is familiar with these models. (Readers unfamiliar with these models are directed to Herrero and Salmeron.[11] For a single input of team size variable, single contextual variable of software size in FP and a single output variable of software effort, we compute the CCR and the BCC inefficiencies for the software projects. We used a software package called Frontier Analyst Profession by Banxia Software Inc. for our analysis. As shown in Figure 3, the software provides the option for running both the CCR and the BCC models.

We test the hypothesis that relationship between team size and software effort is constant returns to scale. We assume non-parametric exponential and half-normal distributions to test our hypothesis. For data where skewness is a large positive number and all the values of variables are positive, half-normal distribution is more preferred distribution. Exponential distribution is a common non-parametric distribution for positive data values. Since all the three variables in our study are positively skewed and all the values take positive values, we test our hypothesis under these two different distributions. The F-ratio (df=1080) value of 1.30 for exponential distribution was significant at p=0.01, and the F-ratio (df=540) value of 1.54 for half-normal distribution was significant at p=0.01.

The null hypothesis of constant returns to scale was rejected. The results indicate the existence of variable returns to scale economy, which means that proportional change in team size will mean higher or lower proportional change in software effort. The constant returns to scale (CRS) means that proportional change in team size will mean same proportional change in software effort. Using the variable returns to scale model instead of the CRS model will lead to improvement in software cost estimation. This improvement can be estimated using a formula provided by Banker and Slaughter.[4] For our dataset this cost reduction formula can be written as noted in the figure here.

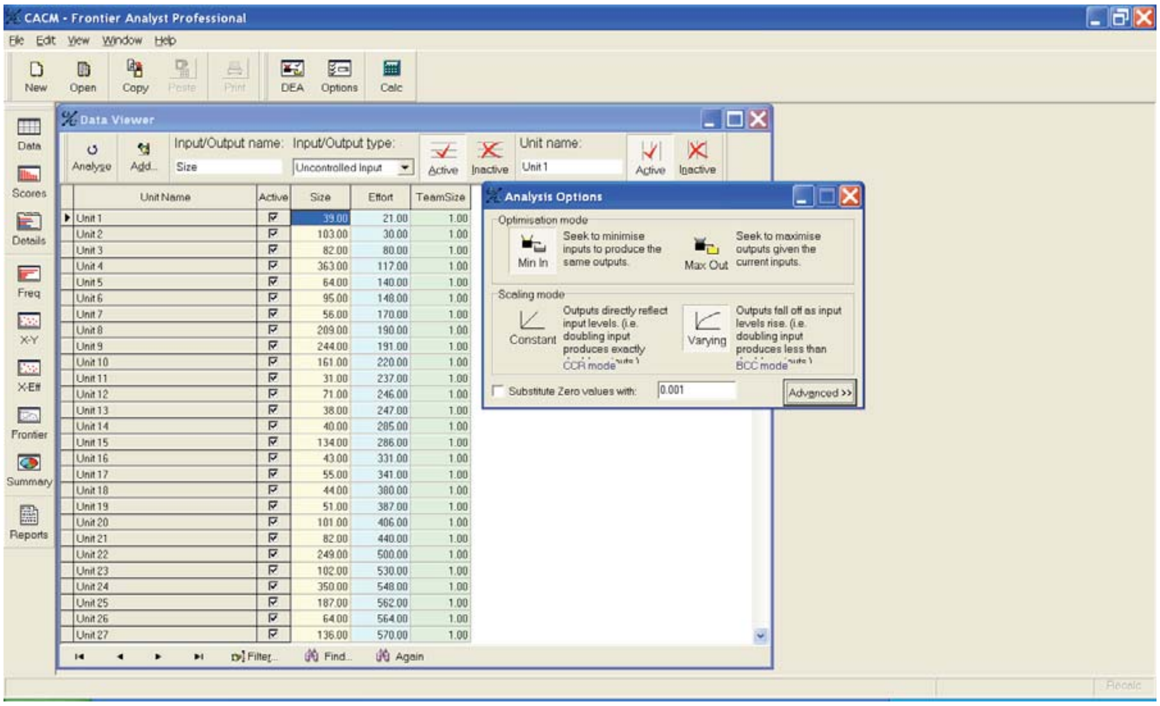
$$Total\ Cost\ Reduction = Wage\ Rate * Total\ Effort \left(1 - \sum_{i=1}^{540}\left(\frac{\theta_i^B}{\theta_i^C} * \frac{Actual\ Effort\ for\ ith\ Project}{Total\ Effort}\right)\right).$$

At a wage rate of \$65/hour and the total effort of 2,817,557 man hours, a maximum cost saving of \$184.15 million can be achieved.

Conclusion

We have investigated the relationship between team size and software cost and found a non-linear variable returns

Figure 3: The DEA data analysis in Frontier Analyst Professional


CACM - Frontier Analyst Professional
File Edit View Window Help
New Open Copy Paste Print DEA Options Calc
Data Scores Details Freq X-Y X-Eff Frontier Summary Reports
Data Viewer
Analyse Add...
Active Inactive
Active Inactive
Filter... Find... Again
Analysis Options
Min In
Max Out
Constant
Varying
Advanced >>
Recalc

Input/Output name: Size
Input/Output type: Uncontrolled Input
Unit name: Unit 1

Unit Name	Active	Size	Effort	TeamSize
Unit 1	✓	39.00	21.00	1.00
Unit 2	✓	103.00	30.00	1.00
Unit 3	✓	82.00	80.00	1.00
Unit 4	✓	363.00	117.00	1.00
Unit 5	✓	64.00	140.00	1.00
Unit 6	✓	95.00	148.00	1.00
Unit 7	✓	56.00	170.00	1.00
Unit 8	✓	209.00	190.00	1.00
Unit 9	✓	244.00	191.00	1.00
Unit 10	✓	161.00	220.00	1.00
Unit 11	✓	31.00	237.00	1.00
Unit 12	✓	71.00	246.00	1.00
Unit 13	✓	38.00	247.00	1.00
Unit 14	✓	40.00	285.00	1.00
Unit 15	✓	134.00	286.00	1.00
Unit 16	✓	43.00	331.00	1.00
Unit 17	✓	55.00	341.00	1.00
Unit 18	✓	44.00	380.00	1.00
Unit 19	✓	51.00	387.00	1.00
Unit 20	✓	101.00	406.00	1.00
Unit 21	✓	82.00	440.00	1.00
Unit 22	✓	249.00	500.00	1.00
Unit 23	✓	102.00	530.00	1.00
Unit 24	✓	350.00	548.00	1.00
Unit 25	✓	187.00	562.00	1.00
Unit 26	✓	64.00	564.00	1.00
Unit 27	✓	136.00	570.00	1.00

Optimisation mode
Min In: Seek to minimise inputs to produce the same outputs.
Max Out: Seek to maximise outputs given the current inputs.

Scaling mode
Constant: Outputs directly reflect input levels. (i.e. doubling input produces exactly CCR mode
Varying: Outputs fall off as input levels rise. (i.e. doubling input produces less than BCC mode

Substitute Zero values with: 0.001

to scale economy exists between team size and software development cost. This seems to indicate that increasing team size does not linearly increase software development cost. The variable returns to scale relationship, indicates that team size should be carefully managed by project managers because its impact is not easy to predict. If there are team communication or coordination problems the managers can assume increasing returns to scale economy which would mean that increasing the team size by certain proportion would require more than proportional increase in resources. Under this assumption, increasing team size for projects facing schedule overrun may not be a good idea. However, if team members are known to efficiently communicate and coordinate their problems then managers can assume decreasing returns to scale economy, where proportional increase in team size would mean less than proportional increase the software cost.

The variable returns to scale relationship between team size and software cost also indicates that when team size variable is used to predict software development cost, using non-linear software prediction models provide a better estimate than using linear regression models. Further, the actual saving resulting from non-linear regression models may be of the order of several million dollars.

Our study does have some limitations that are worth mentioning to the reader. Since we did not screen our projects based on different programming languages, and use of computer aided software engineering tools, our conclusions are not specific to projects that use certain programming languages or CASE tools. The use of object-oriented programming and CASE tools might allow managers to control the cost escalation due to increase in team size. Another limitation of our study is due to unavailable information on team members' experience. The ISBSG data-set does not provide this information. Team member personalities, their experience and knowledge can all impact software development cost.[10] Selecting team members based on their experience and personality type may provide project managers another avenue to control software development cost. The DEA model used in our study has some limitations as well. Since DEA uses extreme points to compute efficiencies, errors or noise in data can have a significant impact on the results. Additionally, for large datasets, linear programming based DEA model can be computationally intensive.

References

1. Angelis, L., Stamelos, I., and Morisio, M. Building a software cost estimation model based on categorical data. *Proceedings of Seventh International Software Metrics Symposium*, London, UK, 2001, 4-15.
2. Banker, R. D., Charnes, A. and Cooper, W. W. Some models for estimating technical and scale inefficiencies in DEA. *Management Science 20*, 9 (1984), 1078-1092.
3. Banker, R. D. and Kemerer, C. F. Scale economies in new software development. *IEEE Transactions on Software Engineering 15*, 10 (1989), 1199-1205.
4. Banker, R. and Slaughter, S. A field study of scale economies in software maintenance. *Management Science 43*, 12 (1997), 1709-1725.
5. Biffl, S. and Gutjahr, W. Influence of team size and defect detection technique on inspection effectiveness. *Proceedings of Seventh International Software Metrics Symposium*, London, UK, 2001, 63-75.
6. Brooks, F. *The Mythical Man-Month.* Addison-Wesley. Reading, MA, 1975.
7. Charnes, A., Cooper, W. W., and Rhodes, E. Evaluating program and managerial efficiency: An application of data envelopment analysis to program follow through. *Management Science 27*, 6, (1981), 668-697.
8. Faraj, S. and Sproull, L. Coordinating expertise in software development teams. *Management Science 46*,12, (2000), 1554-1568.
9. Fried, L. Team size and productivity in systems development. *Journal of Information Systems Management 8*, 3, (1991), 27-41.
10. Gorla, N. and Lam, Y. W. Who should work with whom? Building effective software project teams. *Comm. ACM 47*, 6 (June 2004) 79-82.
11. Herrero, I. and Salmeron, J. L. Using the DEA methodology to rank software technical efficiency. *Comm. ACM. 48*, 1 (Jan. 2005) 101-105.
12. Pendharkar, P.C. and Rodger, J.A. An empirical study of the impact of team size on software development effort, *Information Technology and Management 8* (2007), 253-262.

Parag C. Pendharkar (pxp19@psu.edu) is a professor of information systems at Penn State Harrisburg, PA.

James A. Rodger (jrodger@iup.edu) is a professor of MIS at Indiana University of Pennsylvania, PA.