Financial Pricing of Software Development Risk Factors

Article *in* IEEE Software · November 2010

DOI: 10.1109/MS.2010.28 · Source: IEEE Xplore

2 authors:

Michel Benaroch
Syracuse University
49 PUBLICATIONS **1,081** CITATIONS
SEE PROFILE

Ajit Appari
University of Texas Health Science Center at …
18 PUBLICATIONS **184** CITATIONS
SEE PROFILE

Financial Pricing of Software Development Risk Factors

Michel Benaroch
and
Ajit Appari

Syracuse University
Martin J. Whitman School of Management
Email: mbenaroc@syr.edu

Abstract — The ability to price (monetize) software development risks can benefit various aspects of software development. Cost estimators predict project cost by adjusting the "nominal" cost of a project based on risk factors' (cost drivers') expected values, but the predicted cost often is inaccurate because risk factors' actual values normally deviate from expectations. Since variability is a widely used measure of risk in finance, we present a risk pricing method that relates variability in risk factors to variability in project cost. The method estimates two parameters for each risk factor: extra cost incurred per unit exposure, and project sensitivity, to that factor. We apply the method with COCOMO data and discuss several areas that can benefit from benchmark risk pricing parameters of the kind we obtain.

Keywords — D.2.9.m Risk management, K.6.0.a Economics, K.6.3.a Software Development.

Acknowledgment: The authors thank the Brethen Institute for Operations Research at the Whitman School of Management, Syracuse University, for partial funding.

Introduction

Pricing individual risk factors could benefit software development decision-making because not all software development risk factors are equally critical. For example, suppose that the levels of 'analyst capability' and 'platform experience' assigned to a specific project deviate up or down from what was expected initially. Risk pricing information permits answering such questions as: by how much would the project cost change for one unit deviation of each of these risk factors? and, how much should a project manager be willing to spend to control each of these risks alone? Finance practitioners face similar questions about economic risk factors; for instance: if inflation is one half a percent higher than expected, what would be the expected change in returns on inflation-sensitive assets? To answer such questions, finance relies on risk pricing information comprising the *sensitivity* of asset returns to a risk factor and the *premium-return* expected per unit exposure to that factor. The analogy to software development is straightforward, where asset returns and premium-return are replaced by project costs and premium-cost. A crucial advantage of risk pricing information is that the risk parameters are standardized across

projects: the risk-premium for each risk factor is universal to all projects, and the sensitivity to each risk factor is common to groups of similar projects (e.g., systems software, scientific applications). Hence, these two risk parameters can serve as economic risk benchmark measures.

Risk pricing information is useful in several areas. The *assessment of risk for a portfolio of projects* is one example. Based on a basic principle from insurance, "the only way to successfully manage project risk is to manage risk across your organization's entire project portfolio," because insuring any single project against all its risks would make its price prohibitive.[1] Given risk pricing information, comprising each project's sensitivities to different risk factors and the premium-cost per factor, the portfolio risk is calculated as a function of the sum of smaller project-level risks and the cost of projects:

$$\text{Portfolio Risk} = \sum_{\text{project } i} \left(\sum_{\text{factor } j} \begin{matrix} \text{sensitivity of project } i \text{ to factor } j \\ \text{unit premium-cost for factor } j \end{matrix} \times \right) \times \text{cost of project } i$$

The portfolio risk is expressed as a percentage of projects' cost. For instance, if a project has a 12.3 sensitivity to 'platform volatility' risk, and the premium-cost for 'platform volatility' is 0.3%, then the risk implication is an expected cost premium of 3.69% (=12.3×0.3) on top of the estimated project cost.

Knowing the aggregate portfolio risk and its breakdown by risk factors could enhance various software development decisions. For example, it informs about the economics of platform and process improvement investment decisions, by identifying which risk factors bear more heavily on the project portfolio. It could also benefit software vendors who bid on new projects based on their overall portfolio risk and the respective expected cost, rather than on the risk profile of each new project alone. Likewise, one can imagine how risk pricing information could also benefit the management of risk for a single project.

Before expanding on these and other application areas, we review fundamental risk and risk measurement concepts, and then *present a method for pricing software development risks*.

Literature Review

Risk is usually equated with a possible negative event. If it occurs, it causes a problem that represents a material threat to project outcomes. A risk has an associated occurrence *probability* and *loss* (*impact*). Some fields, such as casualty insurance, are concerned with both the probability and loss, and equate risk with *expected loss*. Other fields, including finance, focus on the distribution of project outcomes due to all foreseeable risk events, and equate risk with the *variance* of this distribution. The 'expected loss' view of risk is prevalent in software development research and practice, but the complementary nature of the

'variance' view of risk is increasingly being recognized.

The 'risk as expected loss' view helped to devise popular risk measures, such as risk exposure (RE), and methods for conducting sensitivity analysis with parametric cost estimators (**Sidebar 1**). The appeal of RE analysis is its simplicity, but it is useful mostly for prioritizing risks (Sidebar 1). Recently RE analysis was also used to develop standardized benchmark risk measures for prioritizing risks.[2] Based on survey data reflecting the experience of project managers on past projects, they quantify the importance of software risks to project performance in terms of their associated occurrence probability and impact. This standardized information can assist an organization in triggering risk-handling activities and in understanding weaknesses of its software development capability. For example, if a risk has a high occurrence probability but low impact, project managers would prefer a strategy that lowers its probability than its impact. However, besides being exposed to the limitations of RE analysis (Sidebar 1), this approach is simply not capable of pricing risks in economic terms.

Parametric cost estimators seek to account for the impact of individual risk factors on software development cost (Sidebar 1). They predict the expected cost of a project based on the expected values of risk factors, which reflect some assumed project characteristics and level of resources. However, cost estimators are not always accurate, in part, because risk factors normally deviate from their expected levels. Consequently, some sensitivity analysis add-on methods use RE analysis to find the range of project costs that may be observed based on likely changes in the values of risk factors (Sidebar 1). However, none of these sensitivity analysis methods is intended or able to price risk factors.

There is a source of uncertainty which can enable pricing risks but has not been considered so far. As we said earlier, risk factors' actual values normally deviate from expectations, causing the actual project cost to deviate up or down from expectation. For example, if a project is expected to be assigned analysts with three years experience but is actually assigned analysts with one year experience, the project cost will increase. We use exactly this kind of information about variations in risk factors and project cost to price individual risk factors, just like finance practice does. By equating *risk* with *variability*, finance applies a method which examines how variability in economic risk factors contributes to variability in asset returns. This method prices each risk factor using two parameters we referred to earlier: *risk premium* and *sensitivity*. We adapt this method to the software development context.

The method we adapt fits well with the 'risk as variance' view, on which work is growing in scale and scope (**Sidebar 2**). Part of this work concerns conventional risk issues such as sensitivity analysis and

project evaluation. A more significant part applies real options methods to risk management issues, especially in the context of agile software development. Agile development methodologies use empirical process control and require to continuously inspect the process[3], but they offer no specific metrics or measurement models for risk management. This is a crucial gap targeted by real options methods with variance-based risk measures (Sidebar 2). In effect, this work and the method we present are in line with value-based software engineering research which adapts financial methods to software development decision-making under risk[4].

Sidebar 1: Work on 'Risk as Expected Loss'

Risk Exposure. If both elements of a risk event – occurrence probability and loss – can be quantified, a common measure of expected loss is *risk exposure*[1]. It is defined as: RE=P(UO)×L(UO), where P(UO) is the probability of an undesirable outcome and L(UO) is loss due to that outcome. For example, if there is a 0.3 probability that the lead project programmer will have to be replaced with a programmer having a lower Language and Tools Experience and the impact is 2 extra person-month (PM) effort, then RE is 0.3*2PM=0.6PM. The appeal of RE analysis is its simplicity and the theoretical premise that RE can be computed to establish a dollar value of risk. However, difficulties in accurately estimating the probability and loss of an unsatisfactory outcome make RE analysis useful mostly for prioritizing risks. Instead of evaluating risks as the product P(OU)×L(OU), a common practice is to map risk on a two-dimensional plane based on the relative ranking of P(OU) and L(OU) and then draw concentric lines to identify risks of the same priority levels[2].

Risk Factors. Some methods approximate the probabilities of undesired outcomes via *risk factors* that can be seen as drivers of those outcomes. These factors fall into categories: process maturity (e.g., use of modern practices, development tools), technological newness (e.g., emerging technology, use of modern development tools), application size and complexity (e.g., scope creep, requirements volatility), development team (e.g., analyst capability, programmer capability), etc. Each of these risk factors represents an uncertain attribute of a project or its contextual environment. The degree of exposure of a project to such risk factors will contribute to the increased probability of occurrence of undesirable outcomes.

Cost Estimators. Parametric models like COCOMO estimate project cost as a function of the expected values of risk factors. Fairley[2] explains how such models are developed. Given data from past projects for relationships of interest (e.g., software size vs. required effort), a linear regression equation is derived in the log-log domain – e.g., log(*Effort*)=log(*a*)+*b*×log(*Size*) (**Figure A**). This equation minimizes the residuals between the equation and the actual projects. Transforming it to the real domain gives the relationship $Effort=a\times Size^b$. In addition, weights can be assigned to these factors to model their effects (**Table A**). The model then has the form: $Effort=(a\times Size^b)\times EAF$, where *EAF* is an effort-adjustment factor or the product of multiplier values from Table A.

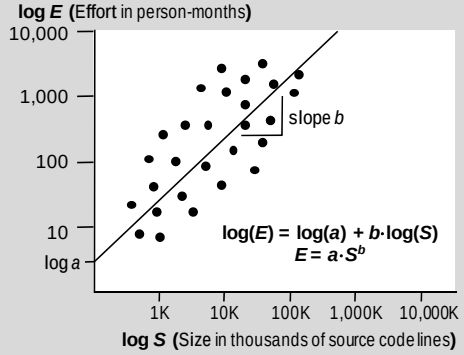

Figure A: regression-based cost estimator

	Effort Multiplier		
Risk Factor (Cost Driver)	Low	Medium	High
Platform Volatility (PVOL)	0.87	1.00	1.15
Application Experience (AEXP)	1.13	1.00	0.91
Product Complexity (CPLX)	0.85	1.00	1.15
Analyst Capability (ACAP)	1.19	1.00	0.86
Prog. Language Experience (PEXP)	1.07	1.00	0.95

Table A: Effort Multipliers for a Software Project

Sensitivity Analysis. Since cost estimators predict cost based on expected values of risk factors, some methods use RE analysis to conduct sensitivity analysis for project cost. RiskTool[3] is one example. Suppose that the estimated COCOMO effort for a project is 50 PM. The manager identifies analyst capability (ACAP) as one of the risk factors. The analyst assigned to the project, Jim, may be transferred to another project, and any replacement analyst would need two PM to be brought to speed on the project. The ACAP rating of Jim is 'high,' with a 0.86 effort multiplier (Table A). Two potential replacement analysts, Dave and Bob, have an equal probability of being hired of 0.5, and their ACAP ratings are 'nominal' and 'low,' with effort multipliers 1.00 and 1.19, respectively. The revised cost estimate becomes 57.8 PM (=50+0.5×2+0.5×(0.5×50×(1.00–0.86)/0.86)+0.5×(0.5×50×(1.19–0.86)/0.86)). Repeating this analysis for all possible risk events and risk factors produces a range of project costs.

Fairley[2] takes this idea further. To assess the probability of a risk item becoming a problem and its effect on project cost, the *a-priori* probability distributions of risk factors (e.g., code size, required memory) are given and Monte-Carlo simulation is run. Monte Carlo simulation rolls a dice for each of the risk factors, with their respective probability distributions, and calculates using a cost estimator the resulting project cost. Repeating this simulation process thousands of times with different input values for risk factors produces the probability distribution of project cost which aggregates the effects of all risk factors.

References

1. B.W. Boehm, "Software Risk Management: Principles and Practices," IEEE Software, 8(1), 1991, 32-41.
2. R. Fairley, "Risk Management for Software projects," *IEEE Software*, 11(3), 1994, 57-67.
3. K. Känsälä, "Integrating Risk Assessment with Cost Estimation," *IEEE Software*, 14, 1997, 61–67.

Sidebar 2: Work on 'Risk as Variance'

One stream of work holding the 'risk as variance' view is concerned with conventional risk issues. One example is sensitivity analysis using Monte-Carlo simulation, which uses the variance of risk factors' probability distributions as a measure of risk (Sidebar 1). Another is the use of variance-type risk analysis to assess the impact of a possible process modification on certain software process attributes[1].

Other work applies real options methods, which relies on variance-based risk measures and financial models. Hakan Erdogmus and his colleagues[2] offer a good review of how these methods have been applied in areas ranging from the evaluation of software initiatives, through component-based development, to agile software development.

The relevance of real options methods to agile development is well summarized in an article titled *Real Options Underlie Agile Practices*[3]. Companies like Yahoo and Microsoft use real options concepts to manage agile development processes that maximize client value[4]. Examples are: (1) the SCRUM and XP practice of *deferring decisions* on which Backlog items to develop until just before coding starts, so that the client can postpone her decision on requirements until she incorporates late breaking information; (2) the *cancel-after-any-phase* practice prioritizes work by business value and allows the customer to halt the project after any phase; and, (3) the *delay implementation of fuzzy features* practice of XP which helps to prioritize requirements at the start of each iteration. Each of these agile practices maps directly to real options concepts[4].

Real options methods not only validate the intuition of agile development methodologies, but also allow formalizing and optimizing the heuristic thinking typifying agile practices. Real options methods with variance-based risk measures can formally answer questions relating to agile practice, for example: Is *passive* deferral of requirements prioritization always optimal? and, Can *proactive* learning-oriented risk mitigations add value? In this respect, Benaroch and Goldstein[5] describe an option-based approach for optimizing risk management practices, including in agile software development.

References

1. S. Beydeda S. and Gruhn V., "Dynamic Evolution of Software Processes to Evolve Software Systems during their Development," *Software Process Improvement and Practice*, 9, 2004, 229-238.
2. H. Erdogmus, J. Favaro, and M. Halling, "Valuation of Software Initiatives under Uncertainty: Concepts, Issues, and Techniques (Chapter 3)," *Value-Based Software Engineering*, S. Biffl et al., eds., Springer-Verlag, 2006.

3. Matts C. and Maassen O., "Real Options Underlie Agile Practices," June 8 2007 (http://www.infoq.com/articles/real-options-enhance-agility)
4. Racheva Z., Daneva M. and Buglione L., "Complementing Measurements and Real Options Concepts to Support Inter-iteration Decision-Making in Agile Projects," *SEAA'08*, 34th Euromicro Conference Software Engineering and Advanced Applications, 2008, 457-464.
5. M. Benaroch and J. Goldstein, "An Integrative Economic Optimization Approach to Systems Development Risk Management," *IEEE Transactions on Software Engineering*, 35, 2009, 638-653.

Risk Pricing Method

We present the risk pricing method in terms of its main underlying principles while avoiding unnecessary technical details in the interest of space and conceptual clarity. The method uses information about variability in risk factors to explain variability in project cost (**Figure 1**). A cost estimator like COCOMO expresses the *expected project cost*, C^E, in monetary (effort) terms, as a function of the *expected* values of risk factors, $(f_1^E,...,f_J^E)$. If the *actual* values of risk factors, $(f_1^A,...,f_J^A)$, deviate from their expected values, the *actual* project cost, C^A, will also deviate from C^E. The difference between actual and expected project cost, C^A–C^E, is mostly due to deviations of risk factors from their expected values and to a statistical error of the cost estimator. However, the effect of the former element dominates. The *relative cost deviation*, $c_i = [C^A - C^E]/C^E$, can therefore be termed the *risk-adjusted cost-premium* percentage for a project. The *relative factor deviation* for risk factor f_j is similarly defined as $d_j = [f_j^A - f_j^E]/f_j^E$.

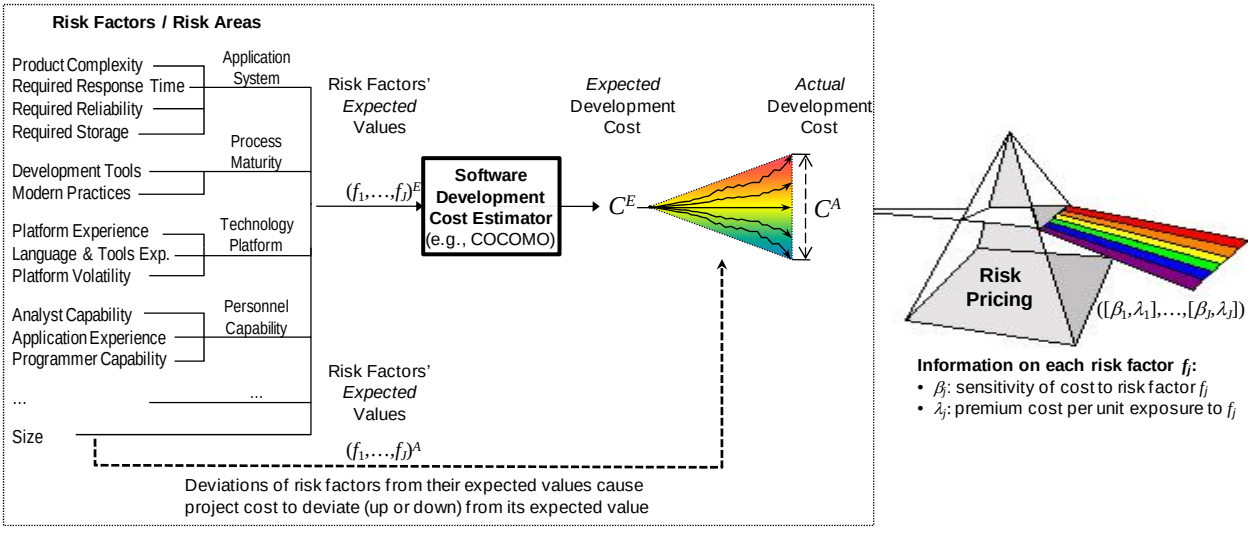

Figure 1: Summary of the approach of the risk pricing method

The risk pricing method makes some assumptions about these notions. First, relative factor deviations are random variables having an expected value of zero. Second, factor deviations are independent or have

been de-correlated; this assumption is common in software development research and in most parametric cost estimators[5,6]. Third, projects naturally fall into groups based on similarities along demographic attributes, such as domain (financial, avionics, etc.) and operating system (Windows, Unix, etc.). Fourth, all agents subscribe to the same (reference) cost estimator. Last, if agents agree on the sources and level of risks in a project, they also hold homogenous expectations about the premium cost that the project would incur in excess of its expected cost. None of these assumptions is restrictive in any practical sense.

The method is anchored in two financial relationships linking the notions of risk and risk-adjusted cost[7]. The first says that the risk-adjusted cost-premium for project *i* in group *g*, c_i, is a linear function of deviations of *J* risk factors:

$$c_i = \beta_0^g + \beta_1^g d_{i,1} + \beta_2^g d_{i,2} + ... + \beta_j^g d_{i,j} + ... + \beta_J^g d_{i,J} + \varepsilon_i \qquad \textbf{(1)}$$

Here, $d_{i,j}$ is the relative deviation of factor *j* for project *i*, β_j^g is the *sensitivity* of the 'average' project in group *g* to factor *j*, β_0^g is the 'base' cost-premium if all factors have zero deviations, and ε_i is idiosyncratic noise. The second relationship says that equation (1) can be transformed into the function:

$$c^g = \lambda_0 + \lambda_1 \beta_1^g + \lambda_2 \beta_2^g + ... + \lambda_j \beta_j^g + ... + \lambda_J \beta_J^g \qquad \textbf{(2)}$$

Here, c^g is the risk-adjusted cost-premium for the 'average' project in group *g*, λ_j is the *cost-premium* for one unit exposure to factor *j* for the 'average' of all projects in all groups, and λ_0 is the cost-premium of a project with zero sensitivities to all factors.

These relationships are useful only once benchmark risk parameters – betas and lambdas – have been derived using historical project data.

Empirical Application of the Method

We employ COCOMO as a reference cost estimator, and use public COCOMO data on 156 projects of varying sizes and types. Since the projects were executed in different organizations, 63 at TRW and 93 at NASA, we took steps to reconcile their differences in consultation with experts in both organizations. Based on their type, or domain, the projects fall into four groups (business applications, support software, system software, and miscellaneous applications).

Project data include demographic variables (e.g., project type, development mode) and the actual cost, software size, and values for COCOMO risk factors. We utilized these data to derive the inputs necessary for pricing risk factors.

- Factor Deviations. Since the data contains only actual values for risk factors, we proxy the expected

value of factor *j* for project *i* by the average of factor *j*'s actual values across all projects in group *g* to which project *i* belongs. This trick works for COCOMO's cost drivers whose value ranges are well bounded. However, it does not work for project size. Consequently, project size is excluded from the present analysis even though it is known to be a crucial risk factor exhibiting variability[5].

- Risk-Adjusted Cost Deviations. Since only the actual project cost is available, the expected project cost is estimated using COCOMO based on: risk factors' expected values, the project's development mode, and the actual values for Size, Schedule, and Turnaround Time.

We took two more steps to satisfy the method's assumptions. First, cost deviations were log transformed to yield a regression-acceptable distribution. Second, factor deviations were de-correlated using statistical factor analysis, similar to numerous other studies involving COCOMO datasets with highly correlated risk factors[6]. Starting with 12 risk factor components, we extracted four usable macro risk factors: *application task*, *process maturity*, *personnel capability*, and *technology platform* (Figure 1). Their values were then computed as averages of their original risk factor components, weighted by factor loadings.

To estimate risk parameters for the four macro risk factors, we used two regressions with a sample of 135 projects. (Other projects were set aside for purposes discussed shortly.) One regression with Equation (1) was run separately for each group, yielding factor sensitivities β_j^g for the 'average' project in each group *g*. The results are summarized in **Table 1**. For the *Application System* factor, sensitivities are positive and statistically significant for all groups. For the *Personnel Capability* factor, sensitivities are significant but negative because upward deviations of personnel capability cause a decline in project cost. For the *Process Maturity* and *Technology Platform* factors, sensitivities are significant for only some groups. The second regression is run once with a slight variation of Equation (2) and only those macro risk factors whose sensitivities are significant for all groups. The factors' risk premia came out positive and statistically significant (Table 1).

	Group-Wide Sensitivities (β)				Risk Premia (λ)
Group	Business Applications	Support Software	System Software	Miscellaneous Software	All Groups
Intercept	-0.173	-0.011	0.325**	0.148**	0.415**
Application System	4.217***	4.013***	6.330***	5.788***	0.219***
Personnel Capability	-8.803***	-10.327***	-10.116**	-11.411***	0.160***
Process Maturity	-2.234	-2.512	12.009**	-0.634	
Technology Platform	7.240	14.364***	37.412***	29.157***	
# Projects	24	51	23	37	135
R-Sqr	42%	23%	55%	69%	13%

Statistically significant at the: *** 1% level ** 5% level

Table 1: estimated Risk Pricing Parameters

Results Interpretation

These results tell us that some project groups have larger sensitivities to certain risk factors, as one would expect. This is most visible for System Software projects. The sensitivity of the 'average' System Software project to Technology Platform risk is more than twice that of the 'average' Support Software project. Moreover, the estimated risk premia indicate that the 'average' project (for all groups) incurs a 21.9% (0.219) and a 16% (0.160) cost-premium for a unit exposure to Application System risk and Personnel Capability risk, respectively. And, on average, 41.5% (0.415) of the cost-premium is due to other risk factors not included in the present analysis (e.g., factors in COCOMO-II but not COCOMO-I).[5] The magnitude of these risk premia may seem large, but recall that it aggregates the affect of several granular component risk factors.

Substituting the results for Application System risk and Personnel Capability risk (Table 1) into equation (2) gives the cost-premium by which the expected of a project cost should be adjusted, up or down, to account for normal deviations of risk factors from expectations. For the System Software group, the risk-adjusted log-transformed cost-premium comes out to be $\ln(1+c)=0.415+0.219\times6.33+0.16\times(-10.116)\cong0.2$, or +20% of the COCOMO predicted cost. (Recall that the cost-premium used in the regression equations is log transformed.) For the Business Applications, Support Software, and Miscellaneous Software groups, the risk-adjusted cost premia are –7%, –30% and –13%, respectively.

Usage of Risk Pricing Information

The important implications of risk pricing information relate to its practical uses. An easy use to illustrate is the fine-tuning of cost estimators' predictions. The risk pricing information we derived implies that normal variability in risk factors makes COCOMO overestimate project cost for some groups and underestimates it for other groups. For example, consider a new project which, given some expected risk factor values at the concept stage, has an expected cost (effort) of 400 PM. Based on the derived risk pricing information, the cost of a System Software project would be adjusted upward by 20% to $400\times(1+0.2)\cong480$ PM, whereas the cost of a Support Software project would be adjusted downward to $400\times(1–0.3)\cong280$ PM.

A curious question is: would a project's predicted cost after adjustment using risk pricing information fall closer to its actual cost? We tried to answer this question using projects we set aside and did not use for parameter estimation. The results are very encouraging. Based on PRED(X%), which measures the number of projects whose predicted cost falls within X% of the actual cost, PRED(10%) improves from

14% to 19%, PRED(20%) from 29% to 37%, and PRED(30%) remains unchanged. Moreover, the range of relative errors (REs) is improved for all groups (**Table 2**). The standard deviation of REs drops by up to 12% for all groups except for one, but even this group's range of REs shifts favorably towards 0%.

	Std Deviation of RE		**Range of RE [Min,Max]**	
Project Group	COCOMO	COCOMO + Risk Pricing Information	COCOMO	COCOMO + Risk Pricing Information
Business Projects	23%	**22%**	[-29%,16%]=45%	[-34%,8%]=**42%**
Miscel. Projects	31%	**27%**	[-21,44%]=65%	[-32%,24%]=**56%**
Support Software	39%	**27%**	[-73%,55%]=128%	[-81%,8%]=**89%**
Systems Projects	24**%**	29%	[-84%,-41%]=43%	[**-81%,-29%**]=52%
All projects	38%	**30%**	[-84%,55%]=139%	[**-81%,24%**]=**105%**

TABLE 2: Performance Evaluation of the Relative Error (RE) measure

Apparently, risk pricing information does convey useful information which could benefit other areas in software development. *Project risk management* is one of the broader areas. In risk management, one goal is to prioritize risk factors by weighing their impact on project cost against their mitigation cost[8]. Risk pricing information permits calculating for each factor its impact on project cost as the product of its sensitivity and premium-cost. Then, dividing this number by the cost of mitigation strategies aimed at specific factors could tell which risk factor is more economical to manage. A similar approach could be used to *assess process improvement initiatives*. First the contribution of each risk factor *j* to the project portfolio risk would be calculated. Now, suppose it is estimated that one process improvement initiative lowers the sensitivity of projects in a specific group *g* to factor *j*, from β_j^g to $\beta_j^{g\prime}$, and another initiative lowers the premium-cost of factor *k* for all projects, from λ_k to λ'_k. Then, calculating the project portfolio risk with and without each initiative can inform about the expected economic impact of that initiative. The calculated project risk is also crucial in *assessing and managing portfolio risk*. It enables identifying over-exposure of the portfolio to particular risks and balancing risk across subgroups of projects in the portfolio, among other things.

Pursuing and experimenting with these and other uses of risk pricing information would more fully illustrate the value of this information. It could eventually open the door for more powerful portfolio and risk management methods from finance.

Future Work and Limitations

Potential aside, though, questions remain about the method's practical value and validation. These are not trivial questions because of the chicken-and-egg problem facing new methods. They cannot be addressed without buy-in from project managers on the method's logic and the *potential* value. This paper's main

goal is to produce the requisite buy-in using the proof of concept we presented. This should hopefully stimulate interest in pursuing follow-up studies on three important issues.

- A relatively simple issue is eliciting project managers' opinions about the method presented and about uses they see for risk pricing information. Ideally, we would have liked to report on such information based a interviews with project managers at the organizations where our data originate (NASA and TRW), for example, to shed more light on why different project types and risk factors have different risk pricing parameter values. Unfortunately, since the data is quite dated, this would be of limited value.
- Another issue is testing the method in realistic settings. In particular, it is necessary to examine whether project managers aided by the method's risk pricing information outperform project managers who rely on judgment alone. This can be checked over the course of several projects, especially since our method follows a "rational" approach while managers often "muddle through" or "improvise" in practice. Another worthy question is whether the method's implementation cost is compensated for by the savings and added value it delivers.
- Also crucial is the generalizability of risk pricing information to organizations other than the ones from which data come. The importance of this issue should be apparent for our own results, which are based on relatively dated data coming from atypical organizations (NASA and TRW).

Addressing the last two issues requires reproducing our results with other data. Ideally, if more organizations make data on their projects available, whether based on COCOMO or any other cost estimator, researchers and special interest groups could estimate more accurate and robust benchmark risk parameters. Replicating our empirical estimation also requires attention to several limitations which we faced and may have compromised the accuracy of our results.

The following are simple strategies for avoiding these limitations. First, create homogenous project groups based on multiple demographic project attributes. Our groups are not very coherent since they were based on a single attribute, project type. Second, use a more recent and larger data set that the one we used, to ensure coverage of wider project varieties. Last, use complete data containing both the expected and actual values for risk factors. Our data contained only actual values, and this forced us to proxy the expected values. Moreover, it forced us to drop one of the more crucial risk factor – project size – which could not be meaningfully approximated. These strategies may appear demanding, but our encouraging results strengthen the belief that the method presented produces valuable risk pricing information.

References

1. B. Kitchehnam and S. Linkman, “Estimates, Uncertainty, and Risk,” *IEEE Software*, 14(3), 1997, 69-74.
2. W. Han and S. Huang “An Empirical Analysis of Risk Components and Performance on Software Projects,” *The Journal of Systems and Software*, 80, 2007, 42-50.
3. Schwaber K. and Beedle M., *Agile Software Development with Scrum*. Prentice Hall, 2001.
4. S. Biffl, A. Aurum, B. Boehm, H. Erdogmus H., and P. Grünbacher, eds., *Value-Based Software Engineering*, Springer-Verlag, 2006.
5. B. Boehm, *Software Cost Estimation with COCOMO*, Prentice Hall, 2002.
6. K. Maxwell, L. Wassenhove, and S. Dutta, “Software Development Productivity of European Space, Military, and Industrial Applications,” *IEEE Trans. Software Eng*., 22(10), 1996, 706-718.
7. J. Elton and M. Gruber, *Modern Portfolio Theory and Investment Analysis*, John Wiley & Sons, 1995.
8. Xu Z. et al., “Optimizing Software Process Based On Risk Assessment and Control”. *Proceedings of International. Conference on Computer and Information Technology*, 2005.