A Practical Approach to Quantifying Risk Evaluation Results

There is a vast literature documenting approaches and tools that address risk assessment and mitigation. In this paper, "hard" and "soft" classifications are introduced, that are based on either the mathematical rigor describing the development of the model or the mathematical rigor expected from the user during the use of the tool. The goal is to present a simple, practical approach to risk analysis, combining the identified benefits, without suffering from the known liabilities. The solution presented here is a combination of the Software Engineering Institute/Software Risk Evaluation (SEI/SRE) method, and Constructive Cost Model (COCOMO).

Software risk management, if practiced properly, is a set of continuous activities for identifying, analyzing, planning, tracking and controlling risks, which is conducted in the context of daily project management. A project planner's first reaction may be to avoid risks all together, but relying strictly on avoidance as a risk mitigation technique is usually inadequate.

Project success primarily depends on the ability to manage the delicate balance of opportunities and risks. Unfortunately, when all risk goes away, so does opportunity. Since risks ultimately manifest themselves incrementally as unexpected cost elements, risk management can also be viewed as a way to dynamically handle the cost/benefit analysis of a project. While techniques for risk identification are usually handled separately from software cost estimation, cost aspects of risks can be used as a communication vehicle during risk prioritization. It has also been determined that parametric cost estimation models are well-suited for risk evaluation [1].

The term *parametric* refers to the fact that the cost is determined via the use of algorithms operating on the parameters of mathematical equations. This structure makes parametric models the prime candidates for carrying out rapid, what-if sensitivity analysis of the cost drivers. Due to their inherent characteristics, nonparametric or nonalgorithmic models, such as expert judgment or estimation by analogy, are not well fitted for sensitivity analysis. This leads to our main proposal, i.e., making the connection between an established risk assessment tool (SRE) and an industry-wide accepted parametric software cost model and estimation tool (COCOMO II).

Risk Management

Based on Barry Boehm's work [2], the risk management steps are outlined in Figure 1.

RISK ASSESSMENT	RISK CONTROL
Risk Identification	**Risk Management Planning**
Checklists	Buying Information
Decomposition	Risk Avoidance
Decision Driver Analysis	Risk Transfer
Assumption Analysis	Risk Reduction
Risk Analysis	Risk Element Planning
Performance Models	Risk Plan Integration
Cost Models	**Risk Resolution**
Network Analysis	Prototypes, Simulations
Decision Analysis	Benchmarks
Quality Factor Analysis	Staffing
Risk Prioritization	Analysis
Risk Exposure	**Risk Monitoring**
Risk Leverage	Milestone Tracking
Compound Risk Reduction	Top-10 Tracking
	Risk Reassessment
	Corrective Action

Figure 1. *Risk Management Steps*

This paper focuses on the connection between software risk identification and cost-model based risk analysis, using risk exposure as a prioritization tool. (The taxonomy based questionnaire, which will be discussed in detail later, is basically a checklist.). Please note that this approach permits the determination of cost ramifications of risks only in the software development domain. Other very quantifiable business risks, such as loss of market opportunity, and loss of sales, can be determined from software development data, but cannot be automatically computed. Similarly, tools can provide quantification of risks, but the overall prioritization and resolution has to be done in the full context of project management.

Risk Taxonomies

Generally defined, software risk taxonomy[1] provides a basis for systematically organizing and analyzing risks in a software project. *Risk Taxonomies*, is intentionally plural, because in addition to describing the importance of a specialized risk taxonomy, we also want to note a level of what we consider undesired proliferation of software risk taxonomies.

Overview of Risk Taxonomy Related Articles

Without assuming completeness, a brief description of current articles follows, where overt or covert development[2] of risk taxonomies plays a role:

- The SEI report lays the foundation of the development of the SEI taxonomy, and discusses the basic concepts of risk taxonomies [3].
- In P.R. Garvey's presentation, the risk elements are described in risk templates, and the taxonomy is implemented via web-based links [4].
- T. Moynihan chose to elicit constructs from experienced managers to determine how they assess risk, after deciding that the taxonomies published in the literature were inadequate [5].
- H. Barki et al. conducted a wide review of the literature and determined 35 variables that were used as taxonomy for risk assessment [6].
- R.J. Madachy developed an extensive rule-based system (identifying 600 risk conditions), where rules were structured around COCOMO cost factors, reflecting an intensive analysis of the potential internal relationships between cost drivers [7].
- K. Känsälä built his tool around 15 risk items he identified as critical, after filtering the data received from 14 selected companies [8].
- E.H. Conrow and P.S. Shishido documented experiences on large projects at TRW, and defined taxonomy consisting of 17 software risk issues [9].
- At Xerox, the SEI-developed taxonomy and the SRE method [10] was evaluated and used in five major projects. While the taxonomy does not provide a complete coverage for all

SEI Software Risk Evaluation Method

The scope of the SRE method is identification, analysis, and preliminary action planning for mitigation. The software risk taxonomy (See Appendix) provides a consistent framework for risk management. It is organized on the basis of three major software risk classes: product engineering, development environment, and program constraints.

Risk elements of these classes are identified at the next level, which are further decomposed into risk attributes. SEI also developed a taxonomy-based questionnaire to carry out structured interviews by an independent assessment team. A sample, customized segment of the TBQ is shown in Figure 2. (The numbering of the questions refers to the original numbering in the full, complete SEI documentation).

Risks are identified and recorded in interviews. After interviews, based on perceived severity and probability of occurrence, the assessment team determines risk magnitude ratings. (Figure 3.)

Note that the relevance of the shaded rows is explained later, when this risk report sample is used to demonstrate the new process. First the assessment team will filter, consolidate and interpret the results. Every risk item is rated separately by the assessment team members (A, B and C in the example), using the risk magnitude matrix (Figure 6.) Severity and probability are separately rated on a scale from one to three, and risk magnitude is computed as severity times probability. This results in a one to nine numerical rating, where one represents improbable and marginal risks, and nine represents risks considered very likely and catastrophic.

Class: A. Product Engineering
Element: 2. Design & Implementation
Attribute: d. Performance

Starter question: [22] Are there any problems with performance?

Cues:
Throughput
Scheduling asynchronous events
Real-time responses
Impact of hardware/software partitioning

Starter question: [23] Has a performance analysis/simulation been done?

Follow-up questions: (YES) (23.a) What is your level of confidence in the results?
(YES) (23.b) Do you have a model to track performance?

Figure 2. *Taxonomy Based Questionnaire Sample*

situations, combining it with a customized SEI Taxonomy-based Questionnaire (TBQ) makes it the preferred tool for assessing risks in the majority of software projects [11].

We found that all authors decided that the introduction of new risk categories, or the creation of a whole new taxonomy, was needed. In our opinion, this is not always justified. During the pilot of the SRE method at Xerox, the SEI taxonomy was criticized in two areas. In large software projects, respondents complained that TBQ terms and language did not always map to local terminology (for example, the classification of contractor relationships). Second, respondents from firmware development projects stated that in their work the distinction between hardware and software was somewhat blurred, and consequently their risk issues were not always adequately covered.

Conclusion Drawn from the Literature Review

It seems that the application of any risk taxonomy always requires a certain level of customization before use, and the quest for the perfect taxonomy, consequently the perfect risk management tool, is fruitless. Also, in the case of actual, computer-based tools, eventually the taxonomy ends up hard-coded into the tool. Instead of further specialization, the approach should be exactly the opposite; we should step back and find a framework that is applicable for a large class of projects, with the understanding that a certain level of customization will take place. As stated earlier, the SEI taxonomy satisfied this requirement.

Cost Estimation with COCOMO II

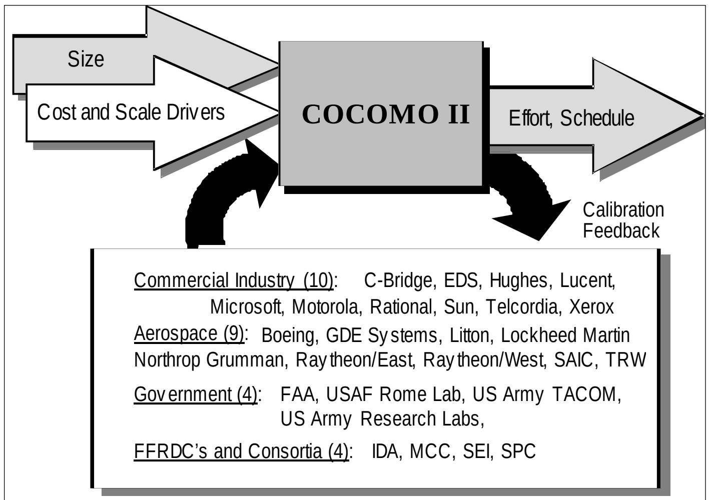
Xerox is interested in the application of COCOMO for software cost estimation, and participates in the University of Southern California/Center for Software Engineering (USC/CSE) Industrial Affiliates Program. At this time 27 industrial affiliates provided data or input to enhance and fine-tune the COCOMO II model (as seen in Figure 4).

Here we provide a conceptual introduction to COCOMO. For up-to-date details, see the appropriate materials on the web site, http://sunset.usc.edu, or in hard copy format [12] for an introduction. (Please note that since publication of that article, the model was renamed to COCOMO II from COCOMO 2.0.)

The COCOMO II model uses 161 data points from affiliates' databases, and is the enhanced version of the earlier COCOMO 81, which was developed using only 64 data points from TRW. Besides refining the model, the university also provides MS/Windows, Sun, and Java versions of the tool, based

Risk Class and Element from Taxonomy Based Questionnaire	Issues and Concerns Recorded during the SRE sessions	Risk Magnitude Rating			
		A	B	C	Team
Program Constraints/Resources	Currrent Plan is schedule driven	6	9	9	8.0
Program Constraints/Resources	Bottom-up plans do not support the schedule	6	9	9	8.0
Development Environment/Management Process	Management is not ready to reconcile the differences between the engineering plan and the business plan	9	6	9	8.0
Program Constraints/Resources	Top-level plan is unrealistic, and it is not based on past track-record and experience	6	6	9	7.0
Development Environment/Management Process	Inability in estimating effort due to the lack of experience with the new technology	4	6	9	6.3
Development Environment/Development Process	Feedback from implementers to architects takes too long, and there is no closure on certain issues	4	9	6	6.3
Development Environment/Development System	Capacity limitaions of the development system (network bandwidth and the speed of compilation) impact schedule	6	6	6	6.0
Program Constraints/Resources	Lack of confidence in the current plans	4	6	6	4.6
Development Environment/Development System	Lack of availibility of adequate number of software licenses	3	4	4	3.6

Figure 3. *Sample Record of Risk Issues during an SRE*


Size
Cost and Scale Drivers
COCOMO II
Effort, Schedule
Calibration Feedback
Commercial Industry (10): C-Bridge, EDS, Hughes, Lucent, Microsoft, Motorola, Rational, Sun, Telcordia, Xerox
Aerospace (9): Boeing, GDE Systems, Litton, Lockheed Martin Northrop Grumman, Raytheon/East, Raytheon/West, SAIC, TRW
Government (4): FAA, USAF Rome Lab, US Army TACOM, US Army Research Labs,
FFRDC's and Consortia (4): IDA, MCC, SEI, SPC

Cocomo II Cost and Scaling Drivers

Cost Drivers

ACAP	Analyst Capability
AEXP	Applications Experience
CPLX	Product Complexity
DATA	Database Size
DOCU	Documentation to match lifecycle needs
LTEX	Language and Tool Experience
PCAP	Programmer Capability
PCON	Personnel continuity
PEXP	Platform Experience
PVOL	Platform Volatility
RELY	Required Software Reliability
RUSE	Required Reusability
SCED	Required Development Schedule
SITE	Multi-site operation
STOR	Main Storage Constraint
TIME	Execution Time Constraint
TOOL	Use of Software Tools

Scaling Drivers

PREC	Precedentedness
FLEX	Development Flexibility
RESL	Architecture/Risk Resolution
TEAM	Team Cohesion
PMAT	Process Maturity

Figure 4. *COCOMO II Software Cost Estimation Model*

Actual Rating	Entry for the tool's screen	Very Low	Low	**Nominal**	High	Very High
Rating Guidelines	Relationship to nominal	75%	85%	**100%**	130%	160%

Figure 5. *COCOMO Rating Guidelines for* ***SCED***

SEVERITY	PROBABILITY		
	Improbable	Probable	Very Likely
Catastrophic	3	6	9
Critical	2	4	6
Marginal	1	2	3

Figure 6. *Risk Magnitude Matrix*

on the current version of the model. Due to the model's popularity, a number of industrial tool vendors incorporated the COCOMO II model into their software cost estimation tool offerings.

Size is the main driver of cost, so the first step of cost estimation is to provide proper size estimation for the project. COCOMO II accepts source line of code (SLOC) and function point input. During the estimation process, the estimator determines the value of scaling constants and cost drivers, using the supplied rating tables, and enters the value in the appropriate screen of the tool. An example based on the COCOMO Model Definition Manual for the cost driver rating guideline is shown in Figure 5.

SCED (required development schedule) belongs to the group of cost drivers that characterize the project to be estimated, and it measures the schedule constraint imposed on the project team. The ratings are defined in terms of percentage of schedule stretch or compression with respect to a nominal schedule for a project requiring a given amount of effort. Compressed or accelerated schedules tend to produce more effort in later phases of development because more issues are left to be determined and resolved.

"Hard" and "Soft" Approaches

As indicated previously, we classify the different risk analysis approaches based on the mathematical rigor required. The first example of a hard approach is offered by Madachy, where he creates the risk taxonomy outright, around the COCOMO cost drivers. This impressive system uses knowledge-engineering methods to refine the risk quantification scheme [7].

In the second example, Känsälä uses Madachy's basic approach, but instead of working around a particular cost estimation tool, he derives his own risk database using risk questionnaires and historical project data. Experimental integration of this risk front-end, *RiskMethod*, was carried out with three different cost estimation tools [8].

The underlying principle in both cases is the use of regression analysis to determine the model's internal coefficients. The approaches require extensive calibration to achieve acceptable results. Finally, the authors' initial objective was not only to assess and prioritize, but also to quantify software risks.

Känsälä also provides the first example of a soft approach. In this TRW approach, the author defines a list of specialized risk issues that could be viewed as a one-level risk taxonomy. This taxonomy is used by internal risk review boards to assess risks via intensive monthly sessions with key representatives of functional and support areas. No particular efforts are made to quantify the impact of identified risks.

Conrad and Shishido give a second example, in which a more complex taxonomy is used by a combined external-internal assessment team in a single assessment. For several days the team formally interviews a cross-section of the development organization for several days. The interviewees are not necessarily key representatives, and they represent a vertical sample of people in the development organization. Nonattribution is a key guiding principle during the sessions, and the names of those raising concerns are kept confidential [9].

	Risk Class and Element from Taxonomy Based Questionnaire	Issues and Concerns Recorded during the SRE sessions	COCOMO Driver Mapping	Team Risk Magnitude Rating
1	Program Constraints/Resources	Current plan is schedule driven	SCED	8.0
2	Program Constraints/Resources	Bottom-up plans do not support the schedule	SCED	8.0
3	Development Environment/ Management Process	Management is not ready to reconcile the differences between the engineering plan and the business plan	SCED	8.0
4	Program Constraints/Resources	Top-level plan is unrealistic, and it is not based on past track record and experience	SCED	7.0
5	Development Environment/ Management Process	Inability in estimating effort due to the lack of experience with the new technology	SCED	6.3
6	Program Constraints/Resources	Lack of confidence in the current plans	SCED	4.6
		Subtotal for	SCED	41.9
		Average for	SCED	6.5

Figure 7. *Worksheet to facilitate mapping*

The common theme in both cases is the use of interviews and guided discussions, with no provisions for risk quantification.

Xerox used the SEI approach, and the following two, main advantages were identified:

- The taxonomy was broad, but also proved to be detailed enough to carry out efficient interviews.
- The nonattributional approach was useful in uncovering well-known risks obvious to the developer community, but due to lack of trust or broken communication, unacknowledged by management.

It had become obvious that the ability to quantify is helpful in prioritizing and presenting the risks to the decision authority.

These experiences led us to recognize that it would be useful to combine the best of both worlds: keep the SEI method as a risk front-end and use COCOMO for quantification. The benefit is that we maintain flexibility and easy customization at the front-end, while using an already calibrated COCOMO model to quantify the results. We did not perceive the benefit of an expert-system approach, because it introduced a complicated and, in our opinion, unnecessary calibration and learning process.

Using COCOMO II To Quantify Software Risk Evaluation Results

On a conceptual level, the task can be phrased as a m:n mapping from the risk taxonomy into the COCOMO scale and cost-driver taxonomy. This complexity resulted in indentifying nearly 600 risk conditions in Madachy's tool. While the precise mapping and the identification of all possible combinations are necessary to create a knowledge-management tool, we found that the goal is never fully accomplished, and customization has to take place before use anyway. The steps of the recommended soft approach are:

1. Carry out an SRE. Do this by using the detailed guidelines of Sisti et. al. [10] To prepare for sessions, the assessment team has to customize the TBQ before and, to some extent, during the interviews to concentrate on the appropriate TBQ subset. An example for customization is shown in Figure 2. *Impact of hardware/software partitioning* was added to the standard TBQ questionnaire to accommodate special Xerox requirements.
2. Map risks into COCOMO. Mark the relevant COCOMO drivers on the worksheets. Figure 7 shows the relevant fragment of the sample risk record, mapped to the SCED cost driver. All the shaded rows on Figure 3 are removed, since they do not map into SCED.
3. Determine baseline estimates. Execute COCOMO estimation with baselined scale and cost factors. For example, COCOMO II would give 16.8 person/month effort estimation for a 5000 SLOC program, where for sake of simplicity nominal values were used for all cost and scale drivers. Please note that this baseline reflects typical, but hypothetical, project conditions fitting the average profile of COCOMO industrial affiliates. For more accurate estimations, the model has to be calibrated with local organizational data, and the drivers have to be set according to the organization's own, original planning conditions.
4. Determine risk-adjusted estimates and compare results. The main objective is to execute COCOMO estimation with risk-adjusted scale and cost factors and determine the difference between baselined and risk-adjusted estimates. In the simple case we present, this means determing the model's sensitivity to SCED cost-driver changes. Figure 8 shows mapping of the risk magnitude into the cost-driver rating.

Note that the mapping is not automatic, and done for all drivers separately, based on their specifics. In case of SCED, for example, the issue from risk identification point of view is forced schedule compression, so there is no point in analyzing effects of a high or very high rating representing a relaxed, instead of compressed, schedule. Applying the risk adjusted value of very low for the SCED cost-driver, COCOMO II at this time would give 21.6 person/month effort estimation for the same 5000 SLOC program. Again, note that all other cost and scale drivers are set to nominal. The demonstrated what-if scenario shows, that in case of a 5000 SLOC program, the impact of a roughly 75-85 percent schedule compression would be a 28 percent increase in effort.

Summary

Risk management is one of the most critical and most difficult aspects of software project management. It is evident, that SEI/SRE as a risk identification method yielded better, more detailed, and more relevant risk items than any input processes

Risk Magnitude from SRE	1	2	3	4	5	6	7	8	9
SCED Rating for COCOMO	NOMINAL		LOW			VERY LOW			

Figure 8. *Mapping Risk Magnitude into Cost Driver Rating*

customarily used with hard risk assessment tools. This is a preferred lightweight approach, because it uses established, familiar, and well-tested tools. Customization and calibration are always needed, even when comprehensive and sophisticated knowledge-engineering-based tools are used. Therefore, we conclude that applying customization effort to existing tools in a lightweight setup is a more efficient approach than purchasing and implementing new, complex tools.◆

References

1. Voldese, I. S. Risk Analysis Using Parametric Cost Models, *Proceedings of the ISPA/SCEA First Joint International Conference*, Toronto, Ontario, Canada, June 16-19, 1998, pg. 1382-1409.
2. Boehm, Barry W. Software Risk Management. *IEEE Computer Press*, 1989.
3. *Taxonomy-Based Risk Identification*. Technical Report CMU/SEI-93-TR-16.
4. Garvey, P.R. A Risk Management Information System Concept for Systems Engineering Applications. Leesburg, Va.. *28th Annual Department of Defense Cost Analysis Symposium*, September 1994.
5. Moynihan, T. Inventory of Personal Constructs for Risk Researchers. *Journal of Information Technology*, Vol. 6, No. 4, 1996.
6. Barki, H., Rivard, S., and Talbot, J. Toward an Assessment of Software Development Risk. *Journal of Management Information Systems*, Vol. 10, No. 2, 1993.
7. Madachy, R.J. Heuristic Risk Assessment Using Cost Factors. *IEEE Software*, May/June 1997.
8. Känsälä, K. Integrating Risk Assessment with Cost Estimation. *IEEE Software*, May/June 1997.
9. Conrow, E.H., and Shisido. P.S. Implementing Risk Management on Software Intensive Projects. *IEEE Software*, May/June 1997.
10. Sisti, F.J., Joseph, S. *Software Risk Evaluation Method, Version 1.0.* Technical Report CMU/SEI-94-TR-19.
11. Hantos, Peter. Experiences with the SEI Risk Evaluation Method. Portland, Oregon. Pacific Northwest Software Quality Conference, 1996.
12. Boehm, Barry W., Clark, B., Horowitz, E., Westland, C., Madachy, R., and Selby, R. COCOMO 2.0 Software Cost Estimation Model. *American Programmer*, July 1996.

Notes

1. "Covert" taxonomy means that the risk sources and their structure are not visible, and they are embedded in the tool. In case of "overt" taxonomy there is an explicit reference to the description of the risk hierarchy.
2. From *Webster's Dictionary*: tax-on-o-my (tak-'sän—me) *n.* 1: the study of general principles in scientific classification.

About the Author

Peter Hantos is department manager at Xerox, in charge of Software Quality Assurance, SPI, Product Quality Assurance, and Reliability. Previously, he was principal scientist of the Xerox Corporate Software Engineering Center, where he developed the corporate software development standard and the software technology readiness processes. He holds a master's and a doctorate degree in electrical engineering from the Technical University of Budapest, Hungary. Dr. Hantos is a senior member of the Institute of the Electrical and Electronics Engineers, and a member of ACM.

Xerox Corporation
701 South Aviation Blvd., MS: ESAE-375
El Segundo, Calif. 90245
Phone: 310-333-9038
Fax: 310-333-8409
E-mail: peter.hantos@usa.xerox.com

Appendix SEI Software Risk Taxonomy

A. PRODUCT ENGINEERING	B. DEVELOPMENT ENV.	C. PROGRAM CONSTRAINTS
1. **Requirements**	1. **Development Process**	1. **Resources**
a. Stability	a. Formality	a. Staff
b. Completeness	b. Suitability	b. Budget
c. Clarity	c. Process Control	c. Schedule
d. Validity	d. Familiarity	d. Facilities
e. Feasibility	e. Product Control	
f. Precendented		
g. Scale		
2. **Design and**	2. **Development System**	2. **Contract**
a. Functionality	a. Capacity	a. Type of contract
b. Difficulty	b. Suitability	b. Restrictions
c. Interfaces	c. Usability	c. Dependencies
d. Performance	d. Familiarity	
e. Testability	e. Reliability	
f. Hardware Constraints	f. System Support	
g. Non-Development Software	g. Deliverability	
3. **Code and Unit Test**	3. **Management Process**	3. **Program Interfaces**
a. Feasibility	a. Planning	a. Customer
b. Testing	b. Project Organization	b. Associate Contractors
c. Coding/Implementation	c. Management Experience	c. Subcontractors
	d. Program Interfaces	d. Prime Contractor
		e. Corporate Management
		f. Vendors
		g. Politics
4. **Integration and Test**	4. **Management Methods**	
a. Environment	a. Monitoring	
b. Product	b. Personnel Management	
c. System	c. Quality Assurance	
	d. Configuration Management	
5. **Engineering Specialties**	5. **Work Environment**	
a. Maintainability	a. Quality Attitude	
b. Reliability	b. Cooperation	
c. Safety	c. Communication	
d. Security	d. Morale	
e. Human Factors		