We've all heard tales of multimillion dollar mistakes that somehow ran off course. Are software projects that risky or do managers need to take a fresh approach when preparing for such critical expeditions?

A Framework for Identifying Software Project Risks

Software projects are notoriously difficult to manage and too many of them end in failure. In 1995, annual U.S. spending on software projects reached approximately $250 billion and encompassed an estimated 175,000 projects [6]. Despite the costs involved, press reports suggest that project failures are occurring with alarming frequency. In 1995, U.S companies alone spent an estimated $59 billion in cost overruns on IS projects and another $81 billion on canceled software projects [6]. One explanation for the high failure rate is that managers are not taking prudent measures to assess and manage the risks involved in these projects.

Mark Keil, Paul E. Cule, Kalle Lyytinen, and Roy C. Schmidt

Advocates of software project risk management claim that by countering these threats to success, the incidence of failure can be reduced [4, 5]. Before we can develop meaningful risk management strategies, however, we must identify these risks. Furthermore, the relative importance of these risks needs to be established, along with some understanding as to why certain risks are perceived to be more important than others. This is necessary so that managerial attention can be focused on the areas that constitute the greatest threats. Finally, identified risks must be classified in a way that suggests meaningful risk mitigation strategies.

Here, we report the results of a Delphi study in which experienced software project managers identified and ranked the most important risks. The study led not only to the identification of risk factors and their relative importance, but also to novel insights into why project managers might view certain risks as being more important than others. Based on these insights, we introduce a framework for classifying software project risks and discuss appropriate strategies for managing each type of risk.

Since the 1970s, both academics and practitioners have written about risks associated with managing software projects [1, 2, 4, 5, 7, 8]. Unfortunately, much of what has been written on risk is based either on anecdotal evidence or on studies limited to a narrow portion of the development process. Moreover, no systematic attempts have been made to identify software project risks by tapping the opinions of those who actually have experience in managing such projects. With a few exceptions [3, 8], there has been little attempt to understand the relative importance of various risks or to classify them in any meaningful way. Thus, there is a need for a more systematic investigation to identify the major risks that can impact software projects, to classify these risks, and to develop appropriate strategies for each class of risk.[1]

In terms of previous efforts to identify risk factors, Boehm's work [4] has probably had more influence on the practitioner community than any other. Boehm's "top 10 list of software risk items" was built upon his experiences in the defense industry in the 1980s. The projects and environments that were the basis of Boehm's work might not, however, be representative of those in typical business enterprises. Furthermore, both the organizational and technological landscape has changed considerably since Boehm's work appeared; new organizational forms and systems development environments have emerged, new mechanisms have evolved for acquiring systems (for example, outsourcing and strategic alliances), and the centralized, mainframe-centric, systems architecture has given way to distributed computing. For all of these reasons, we judged it to be an appropriate time to reexamine the risk issue.

Our study was designed to address two basic questions: What are the

JASON SCHNEIDER

[1] In decision theory a risk may lead to either positive or negative consequences. Although Charette [5] defines software risk along decision-theoretic lines, the bulk of software risk management literature traditionally has focused on the negative outcomes [2]. Consistent with the focus on negative outcomes, we define a risk factor as a contingency that constitutes a serious threat to the successful completion of a software development project.

factors that software project managers perceive as risks and which of these factors do they consider most important? Can the risk factors be categorized in such a way as to provide insight into appropriate risk mitigation strategies?

To address these questions we assembled panels of experienced software project managers in different parts of the world—Finland, Hong Kong, and the U.S.—and asked them to first identify specific risk factors and then rank and rate them in terms of their importance.[2]

Two very interesting results emerged from the study. In terms of identifying and rating risk factors, there were nearly a dozen factors that all three panels viewed as important, suggesting the existence of a universal set of risks with global relevance. Interestingly, risks that were viewed to be most serious were often those seen as being outside the direct control of the project manager. This observation provided us with a means of categorizing the risk factors in a useful way. Accordingly, to address the second question we introduce a framework based on two dimensions: *perceived level of control*, and *perceived relative importance of the risk*. We then speculate on possible strategies for managing each type of risk.

Perhaps the most interesting finding is that three independent panels, representing very different countries and cultures, selected a common set of 11 risk factors as being among the more important items. While there were differences across panels in the level of importance ascribed to some of these risk factors, the fact that all three panels independently selected these 11 risk factors suggests they are, in some sense, universal. In experimental terms, each panel represents a replication of the same Delphi experiment, adding credibility to our findings.

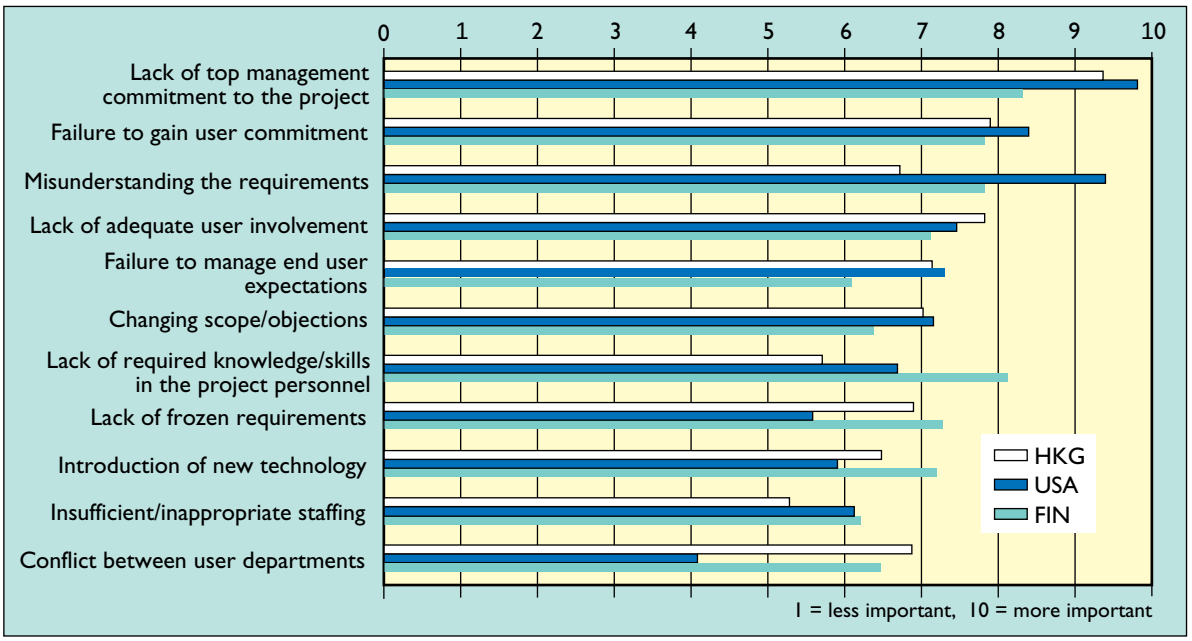

Figure 1. Risk factors identified by all three panels ordered by relative importance

Figure 1 shows the mean importance rating ascribed to each of the 11 risk factors for each country panel. The risk factors are ordered in decreasing level of importance (averaging across the three panels) from top to bottom.

Interestingly, only one of the 11 risk factors that were viewed as important by all three panels involved technology. This item—the introduction of new technology—was not rated particularly high relative to the other items. One reason for this may be that software project managers feel they can control the risks posed by new technology. As one project manager observed: "I expect that risk assessments concerning the technology are fully understood, and provided for in the project plan."

Throughout the Delphi process, panelists shared their insights regarding not only which risks are most important, but why. While space does not permit a full discussion of each risk factor, we discuss some of the panelists' explanations for the importance of the top three risk factors: lack of top man-

[2]We chose an international approach to identify a broader set of risk factors and explore possible cultural differences relating to the identification and ranking of specific risk factors. While some differences were observed across the different panels, this focuses on the factors that were common to all three panels. Readers interested in other aspects of the study should refer to [12].

agement commitment to the project, failure to gain user commitment, and misunderstanding the requirements.

Although the importance of top management support is obvious, it is interesting to note that our panelists chose the term "commitment" (rather than "support") to indicate the strong, active role top management must play in the project from initiation through implementation. Many panelists saw the lack of top management commitment as a risk that overshadowed all others. In the words of one panelist, "if this is not present, then all other risks and issues may be impossible to address in a timely manner."

Another prime area of concern to the panelists was the failure to gain user commitment which was viewed as critical because it helps ensure that users are actively involved in the requirements determination process, and it creates a sense of ownership, thereby minimizing the risk that the system will be rejected. To some, strong user commitment was seen as something that could even compensate for a lack of executive commitment. This remark offered by one of the panelists was typical: "The users of the system to be delivered are the ultimate customer of the deliverable . . . If the users are not committed to a joint effort in which they are heavily involved in the effort, there is a high risk of assuming their detailed functional and business requirements. Without their commitment, they withdraw critical feeling of ownership and the project has a high chance of missing the mark. Even executive commitment lacking can be overcome by total customer/user commitment."

Misunderstanding the requirements was also viewed as a critical risk factor because, in the words of one panelist, "requirements drive the entire project." Without a proper systems analysis to develop a complete and accurate set of requirements there is a distinct possibility of building a system that no one wants to use. For this reason, many panelists underscored the importance of understanding the requirements.

A Risk Categorization Framework

One of the more intriguing findings from our study is that risks thought to be most important are often not under the project manager's direct control. When asked to explain why they perceived certain risk factors to be more important than others, many panelists indicated their perceptions of risk were higher for those items over which they had little or no control. As one panelist remarked: "The reason I ranked customer risk so high is because I cannot effectively *control* user behavior. I cannot tell my customer's departmental vice presidents to hold hands and sing *We Are The World*, when that is what the project desperately needs."

The notion of control and its relationship with perceived importance enabled the development of a framework for mapping different types of risk.

We mapped the different types of risk identified by our panelists into a 2 x 2 grid (Figure 2). One dimension of the grid is perceived importance, which we define as the relative importance of a particular risk factor in relation to other risk factors. Importance is some combination of risk frequency (that is, how likely it is that the risk will occur) and risk impact (such as, how serious a threat the risk represents if it does occur). The second dimension, perceived level of control, represents the degree to which the project managers perceived that their actions could prevent the risk from occurring. Both dimensions are obviously continuums which, for the sake of simplicity, have been reduced to a grid.

We first mapped the individual risk factors into the upper or lower half of the grid using the average perceived importance scores provided by the panelists.[3] As a gauge for assessing the perceived level of control over risk factors (whether an item should be mapped to the left or right side of the grid), we relied on both the panelists' comments and our own experience in managing projects. Once the factors were mapped in this manner, it was possible to derive the broader category names shown in Figure 2. Note that the numbers in the quadrants are for reference only, and do not imply any sort of sequence. Rather, the risks in all four quadrants must be actively managed throughout the entire project.[4] Each of the four quadrants requires a different approach toward managing risk.[5]

Quadrant 1: Customer Mandate. Many of the panelists' top risks fall in this quadrant. The name for this quadrant captures the notion that successful

[3]Risks rated an average of 7 or higher on our 10-point scale were classified as high importance. Risks rated below 7 were judged to be of moderate importance. The term "moderate" was chosen because all of the remaining risks were rated 5 or higher. The mapping was initially performed with the 11 risk factors common to all three panels. Additional factors viewed to be important by one or more of the three panels were then mapped to populate the quadrants more fully and to determine if the framework had face validity.

[4]Interactions undoubtedly exist across quadrants, but because of space limitations these will not be discussed here.

[5]The risk mitigation strategies we discuss are somewhat speculative at this point and are offered for illustrative purposes. More data is needed to determine how effective these, or other, risk mitigation strategies would be in practice.

projects are very often those that have the commitment of both senior management and those who will actually use the system. Without a clear charter, or mandate, the project is simply not viable. Relevant questions for project managers to ask are: Does this project have senior management commitment? Does it have user commitment? In short, is there a clear charter or mandate for me to complete the project?

Examples of specific risk factors in this quadrant include a lack of top management commitment, failure to gain user commitment, and inadequate user involvement. All were viewed as critical items over which the project manager had comparatively little control.

Quadrant 1 requires risk mitigation strategies that create and maintain good relationships with customers and promote customer commitment to the project. Such relationships cannot be built overnight. Therefore, to successfully counter Quadrant 1 risks, both IS senior management and project

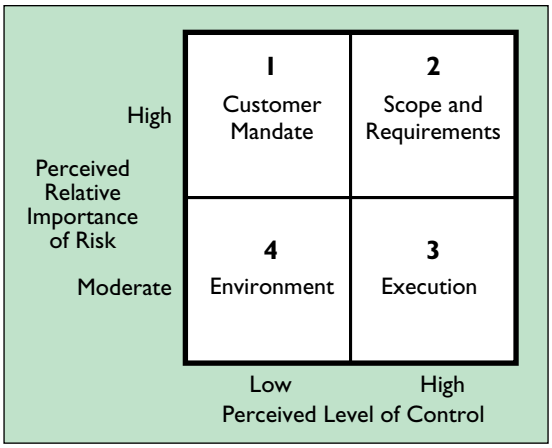

Figure 2. A risk categorization framework

managers must establish and maintain long-term relationships with both their users/customers and with company senior management. An essential element to this relationship building is the project manager's need to build and maintain trust with the users by meeting commitments.

Projects in which either top management or user commitment is lacking represent a high-risk proposition. But, initial commitment is not enough. Once a project has started, project managers must periodically gauge the level of commitment from both top management and the user community to avoid being caught in a situation where support for the project suddenly evaporates.

One approach for establishing and maintaining commitment within the context of a specific project is the application of "Theory-W" [3], which involves structuring the project to meet the "win" conditions of various stakeholders. Projects that enjoy broad-based support across multiple stakeholders are less risky than those narrowly aimed at gaining the commitment of just one stakeholder.

Research also suggests a number of other tactics by which commitment to a project can be fostered [9], including emphasizing the large payoffs associated with successfully completing the project, creating opportunities for senior managers to publicly display their support for the project, and aligning the project with other goals that are viewed as central to the organization.

In addition to the approaches already outlined, managing the risks associated with Quadrant 1 also involves managing end-user expectations. Problems with user acceptance can occur whenever user expectations are not realistic.

To summarize, the risks in this quadrant cannot be controlled by the project manager, but they can be influenced. Project managers must take reasonable steps to ensure that they have the support and commitment needed to deliver a successful project [10]. The risk mitigation for Quadrant 1 involves relationship management, trust-building, and political skills. Project managers must have these skills in order to effectively address the risks in this quadrant.

Quadrant 2: Scope and Requirements. As our name for this quadrant suggests, many of the risks threatening software projects involve the ambiguities and uncertainties that arise in establishing the project's scope and requirements. Relevant questions for project managers to ask are: What is inside the scope of the project and what is outside the scope of the project? What functionality is essential to be successful versus "nice to have?" In short, do I know what I am building and how this might change over time?

Examples of specific risk factors in this quadrant include misunderstanding requirements and not managing change properly. Both are viewed as critically important items, but ones over which the project manager has considerable control.

Quadrant 2 risk mitigation strategies should emphasize the management of ambiguity and change. More often than not, it is impossible to pin down the exact requirements at the outset of a project, hence the popularity of various evolutionary approaches toward system development such as the spiral model [3]. As time progresses, the scope and requirements should become clearer and the users should develop more realistic expectations of what the system will do. One tactic that is helpful in establishing the scope of a project is to specify what

will *not* be included in the project. To avoid the common problem of scope creep, project managers should educate the user/customer on the impact of scope changes in terms of both project cost and schedule. To further guard against Quadrant 2 risks, project managers must be willing to draw a line between desirable and absolutely necessary functionality. Techniques like multicriteria decision-making methods and function-point analysis are useful in such exercises.

Another useful tactic for combating Quadrant 2 risks is to ensure that the project is being driven by the user community and not the developers [8]. Users/customers should be continually reminded of the important role that they play in defining the system's functionality. As one project manager put it: "[I tell them:] You own that application layer—whatever you need it to look like and however you need it to function that's yours and if you don't tell me I can't make it. I don't know your job. You have to tell me how it works."

To summarize, the risks in this quadrant can be largely controlled by the project manager, but do require skillful interfacing with the user/customer. Effective processes for managing change and ambiguity are needed. The real danger associated with Quadrant 2 is that project managers may overestimate their own abilities or fail to realize their own limitations. In either case, this can lead to an underestimation of the risks associated with this quadrant or a failure to develop appropriate risk mitigation strategies.

Quadrant 3: Execution. The risks in Quadrant 3 concern the actual execution of the project. Relevant questions for project managers to ask are: Can the complexities of the system development and implementation be managed successfully? Do I have enough people with the requisite skills and knowledge? In short, given what I know about the project scope and requirements, do I have a team in place that can successfully execute this project?

Examples of specific risk factors in this quadrant include many of the traditional pitfalls associated with poor project management [3]. Issues of inappropriate or insufficient staffing, lack of effective development process methodology, poor estimation, and improper definition of roles and responsibilities can all be located in this quadrant. In general, the experienced project managers who served as our panelists believed they had reasonable control over these risks, and hence they regarded them as moderate rather than high-risk items.

The risk mitigation strategy for Quadrant 3 should emphasize internal evaluations coupled with external reviews to keep a project on track. Tactics include using disciplined development processes and methodologies to break the project down into manageable chunks, clearly defining roles and responsibilities, and developing contingency plans to cope with staffing shortfalls and new technologies. That is, to successfully counter Quadrant 3 risks, project managers must follow an established development methodology and proactively anticipate and respond to events that can threaten the development process.

It is important to note that under normal circumstances, Quadrant 3 risks should not pose a serious threat to the project (hence, their low perceived importance). Basically, risks that fall in this quadrant are generally within the project manager's realm of control. But project managers cannot afford to become complacent in their handling of these risks. Failure to manage the risks in this quadrant can result in poor quality software that is delivered late and over budget (if it is indeed delivered at all).

Quadrant 4: Environment. Risks in this quadrant can be traced to the project environment that exists both inside and outside the organization. While our panelists focused their attention on internal risks, this quadrant could include risks in the external environment (for example, natural disasters or changes in the competitive environment). Relevant questions for project managers to ask are: What are the things that can happen inside the organization that would threaten this project? What are the external events that could threaten the project? In short, what are the things that can go wrong that we haven't addressed in the other quadrants?

Examples of specific risk factors in this quadrant include changing scope/objectives (due to changes in senior management or the business itself), and conflicts that may arise between user departments.

The risks in Quadrant 4 are those over which the project manager has little or no control. They have a low likelihood of occurrence and are, therefore, not viewed as being terribly important. Yet, when they hit a project, they can be significant and dangerous. Quadrant 4 risks are among the most difficult to anticipate and plan for because there is a dearth of

experience in dealing with them. Contingency planning, including concepts and tactics associated with disaster planning, is the most sensible strategy for dealing with Quadrant 4 risks. Scenarios represent another approach for developing plans to deal with these risks should they occur.

Conclusions

Using a systematic approach, we tapped the experience of more than 40 software project managers from around the globe to identify a universal set of risk factors. The three most important risk factors were judged to be a lack of top management commitment to the project, a failure to gain user commitment, and a misunderstanding the requirements. These and the other identified risk factors can serve as a useful checklist for conducting risk assessments of software projects [12].

One of the most interesting findings from this study is the fact that the risks perceived to be most important often lie outside the direct control of the project manager. Most panelists indicated their perceptions of risk were higher for those items over which they had little or no control. Based on this observation, we developed and presented a typology of project risk factors and used this to suggest possible risk mitigation strategies. One of the strengths of this approach is that instead of focusing on individual risk factors, it provides a higher-level framework for thinking about four distinct types of software project risk (and different strategies for addressing each type).

In comparing our results to Boehm's, we find that some of the most important risks identified by our panelists (such as, risks relating to customer mandate) are missing entirely from Boehm's top 10 risk list [4]. Instead, Boehm's list (and much of the existing software project risk literature) focuses on what we have labeled "execution" risks. One explanation for this may be that Boehm and others chose to focus on those risk factors over which the project manager has

How the Study was Conducted

This study employed a variation on the traditional Delphi survey approach designed to elicit opinions from a panel of experts through iterative, controlled feedback. Three expert panels were formed by recruiting from among experienced project managers in each country. A total of 45 software project managers were initially recruited for the study and 41 of these individuals chose to participate in all phases of the research; 19 on the USA panel, 13 on the Finnish panel, and 9 on the Hong Kong panel. On average, panelists had 16 years of work experience and had managed more than 20 software projects.

We adopted a method for the Delphi survey developed by Schmidt [11] which provides a statistical measure of consensus within the panel and allows comparisons to be made across multiple panels. The Delphi survey consisted of three phases. In the first phase, a brainstorming round was conducted to elicit as many risk factors as possible from each of the panels. The output of all three panels was then consolidated to produce a list of 53 risk factors. The consolidation process ensured the three panels had access to a common list of factors with common definitions. During subsequent phases, the panels were allowed to operate independently, each one selecting (and later ranking and rating) the most important risk factors from this common pool of factors.

The purpose of the second phase was to narrow the list of items to a manageable number that could be meaningfully ranked and rated. This was done by having each panelist select the 20 risks deemed to be most important. For each country panel, risk factors that were selected by 50% or more of the panelists in that country were retained for the next phase of the study. The risk factors retained included a set of 11 risk factors common to all three panels.

The third and final phase of the study involved the actual ranking and rating of risk factors. Ranking and rating rounds were conducted until each panel reached an acceptable level of consensus. Kendall's Coefficient of Concordance [11] was used to measure the degree of consensus among panelists for each country. Rounds of ranking were repeated until either the panelists reached strong consensus or consensus did not change from one round to the next.

a relatively high degree of control. One implication of our study, however, is that project managers should not restrict their attention to project execution risks. Instead, they should determine if they have the support and commitment to carry out the project (Quadrant 1); manage the ambiguity and change associated with establishing system scope and requirements (Quadrant 2); select a risk-driven execution strategy (Quadrant 3); and be able to anticipate and respond to unexpected changes in the environment (Quadrant 4). Accordingly, the framework presented here encourages managers to explore a broader set of factors in performing risk assessments. Looking to the future, the effectiveness of different strategies for managing each type of risk needs to be carefully assessed. C

REFERENCES

1. Alter, S. and Ginzberg, M. Managing uncertainty in MIS implementation. *Sloan Manage. Rev. 20,* 1 (1978), 23–31.
2. Barki, H., Rivard, S., and Talbot, J. Toward an Assessment of Software Development Risk. *J. Manage. Info. Syst. 10*, 2 (1993), 203–225.
3. Boehm, B. and Ross, R. Theory-W software project management: principles and examples. *IEEE Trans. Softw. Eng. 15*, 7 (1989), 902–916.
4. Boehm, B.W. Software Risk Management: Principles and Practices. *IEEE Softw. 8*, 1 (1991), 32–41.
5. Charette, R.N. *Software Engineering Risk Analysis and Management.* Intertext, New York, 1989.
6. Johnson, J. Chaos: The dollar drain of IT project failures. *Applic. Dev. Trends 2,* 1 (1995), 41–47.
7. Jones, C. *Assessment and Control of Software Risks.* Prentice-Hall, Englewood Cliffs, N.J., 1994.
8. McFarlan, F.W. Portfolio approach to information systems. *Harvard Bus. Rev. 59*, 5 (1981), 142–150.
9. Newman, M. and Sabherwal, R. Determinants of commitment to information systems development: A longitudinal investigation. *MIS Q. 20*, 1 (1996).
10. Sauer, C. Understanding support—Lessons from a case study. *Australian J. Info. Syst. 1*, 1 (1993), 63–74.
11. Schmidt, R.C. Managing Delphi surveys using nonparametric statistical techniques. *Decision Sciences, 28* 3 (Summer 1997), 763–774.
12. Schmidt, R.C., Lyytinen, K., Keil, M., and Cule, P. Identifying Software Project Risks: An International Delphi Study. Hong Kong University of Science and Technology Working Paper (1997).

MARK KEIL (mkeil@gsu.edu) is an associate professor in the Department of Computer Information Systems at Georgia State University.
PAUL CULE (culep@biz.mu.edu) is assistant professor of Management in the College of Business Administration at Marquette University, Milwaukee, Wis.
KALLE LYYTINEN (kalle@jytko.jyu.fi) is a professor in Information Systems at the University of Jyväskylä, Finland.
ROY SCHMIDT (roy@bradley.edu) is an assistant professor in the Foster College of Business Administration at Bradley University, Peoria, Ill.