#### CHAPTER

# 6

## RISK ANALYSIS AND MANAGEMENT

#### KEY CONCEPTS

assessment 154
components and
<b>drivers</b> 149
$\textbf{identification} \ \dots \ 148$
mitigation 156
monitoring 157
projection 151
refinement 156
risk exposure153
risk strategies 146
risk table 151
<b>RMMM plan</b> 159
safety and
hazards 158

n his book on risk analysis and management, Robert Charette [CHA89] presents a conceptual definition of risk:

First, risk concerns future happenings. Today and yesterday are beyond active concern, as we are already reaping what was previously sowed by our past actions. The question is, can we, therefore, by changing our actions today, create an opportunity for a different and hopefully better situation for ourselves tomorrow. This means, second, that risk involves change, such as in changes of mind, opinion, actions, or places . . . [Third,] risk involves choice, and the uncertainty that choice itself entails. Thus paradoxically, risk, like death and taxes, is one of the few certainties of life.

When risk is considered in the context of software engineering, Charette's three conceptual underpinnings are always in evidence. The future is our concern—what risks might cause the software project to go awry? Change is our concern—how will changes in customer requirements, development technologies, target computers, and all other entities connected to the project affect timeliness and overall success? Last, we must grapple with choices—what methods and tools should we use, how many people should be involved, how much emphasis on quality is "enough"?

### FOOK FOOK

What is it? Risk analysis and management are a series of steps that help a software team to

understand and manage uncertainty. Many problems can plague a software project. A risk is a potential problem—it might happen, it might not. But, regardless of the outcome, it's a really good idea to identify it, assess its probability of occurrence, estimate its impact, and establish a contingency plan should the problem actually occur.

Who does it? Everyone involved in the software process—managers, software engineers, and customers—participate in risk analysis and management.

Why is it important? Think about the Boy Scout motto: "Be prepared." Software is a difficult undertaking. Lots of things can go wrong, and frankly, many often do. It's for this reason that being prepared—understanding the risks and taking proactive measures to avoid or manage them—is a key element of good software project management.

What are the steps? Recognizing what can go wrong is the first step, called "risk identification."

Next, each risk is analyzed to determine the likelihood that it will occur and the damage that it will do if it does occur. Once this information is established, risks are ranked, by probability and impact. Finally, a plan is developed to manage those risks with high probability and high impact.

What is the work product? A risk mitigation, monitoring, and management (RMMM) plan or

#### QUICK LOOK

a set of risk information sheets is produced.

How do I ensure that I've done

it right? The risks that are analyzed and managed should be derived from thorough study of

the people, the product, the process, and the project. The RMMM should be revisited as the project proceeds to ensure that risks are kept up to date. Contingency plans for risk management should be realistic.

Peter Drucker [DRU75] once said, "While it is futile to try to eliminate risk, and questionable to try to minimize it, it is essential that the risks taken be the right risks." Before we can identify the "right risks" to be taken during a software project, it is important to identify all risks that are obvious to both managers and practitioners.

#### 6.1 REACTIVE VS. PROACTIVE RISK STRATEGIES

Reactive risk strategies have been laughingly called the "Indiana Jones school of risk management" [THO92]. In the movies that carried his name, Indiana Jones, when faced with overwhelming difficulty, would invariably say, "Don't worry, I'll think of something!" Never worrying about problems until they happened, Indy would react in some heroic way.

Sadly, the average software project manager is not Indiana Jones and the members of the software project team are not his trusty sidekicks. Yet, the majority of software teams rely solely on reactive risk strategies. At best, a reactive strategy monitors the project for likely risks. Resources are set aside to deal with them, should they become actual problems. More commonly, the software team does nothing about risks until something goes wrong. Then, the team flies into action in an attempt to correct the problem rapidly. This is often called a *fire fighting mode*. When this fails, "crisis management" [CHA92] takes over and the project is in real jeopardy.

A considerably more intelligent strategy for risk management is to be proactive. A proactive strategy begins long before technical work is initiated. Potential risks are identified, their probability and impact are assessed, and they are ranked by importance. Then, the software team establishes a plan for managing risk. The primary objective is to avoid risk, but because not all risks can be avoided, the team works to develop a contingency plan that will enable it to respond in a controlled and effective manner. Throughout the remainder of this chapter, we discuss a proactive strategy for risk management.

#### 6.2 SOFTWARE RISKS

Although there has been considerable debate about the proper definition for software risk, there is general agreement that risk always involves two characteristics [HIG95]:

Quote:

'If you don't actively attack the risks, they will actively attack you."

Tom Gilb

- *Uncertainty*—the risk may or may not happen; that is, there are no 100% probable risks.<sup>1</sup>
- Loss—if the risk becomes a reality, unwanted consequences or losses will occur.

When risks are analyzed, it is important to quantify the level of uncertainty and the degree of loss associated with each risk. To accomplish this, different categories of risks are considered.

What types of risks are we likely to poten tome! software is built?

*Project risks* threaten the project plan. That is, if project risks become real, it is likely that project schedule will slip and that costs will increase. Project risks identify potential budgetary, schedule, personnel (staffing and organization), resource, customer, and requirements problems and their impact on a software project. In Chapter 5, project complexity, size, and the degree of structural uncertainty were also defined as project (and estimation) risk factors.

Technical risks threaten the quality and timeliness of the software to be produced. If a technical risk becomes a reality, implementation may become difficult or impossible. Technical risks identify potential design, implementation, interface, verification, and maintenance problems. In addition, specification ambiguity, technical uncertainty, technical obsolescence, and "leading-edge" technology are also risk factors. Technical risks occur because the problem is harder to solve than we thought it would be.

Business risks threaten the viability of the software to be built. Business risks often jeopardize the project or the product. Candidates for the top five business risks are (1) building a excellent product or system that no one really wants (market risk), (2) building a product that no longer fits into the overall business strategy for the company (strategic risk), (3) building a product that the sales force doesn't understand how to sell, (4) losing the support of senior management due to a change in focus or a change in people (management risk), and (5) losing budgetary or personnel commitment (budget risks). It is extremely important to note that simple categorization won't always work. Some risks are simply unpredictable in advance.

Another general categorization of risks has been proposed by Charette [CHA89]. *Known risks* are those that can be uncovered after careful evaluation of the project plan, the business and technical environment in which the project is being developed, and other reliable information sources (e.g., unrealistic delivery date, lack of documented requirements or software scope, poor development environment). *Predictable risks* are extrapolated from past project experience (e.g., staff turnover, poor communication with the customer, dilution of staff effort as ongoing maintenance requests are serviced). *Unpredictable risks* are the joker in the deck. They can and do occur, but they are extremely difficult to identify in advance.

#### \_ uote:

'[Today,] no one has the luxury of getting to know a task so well that it holds no surprises, and surprises mean risk."

**Stephen Grey** 

<sup>1</sup> A risk that is 100 percent probable is a constraint on the software project.

#### 6.3 RISK IDENTIFICATION

*Risk identification* is a systematic attempt to specify threats to the project plan (estimates, schedule, resource loading, etc.). By identifying known and predictable risks, the project manager takes a first step toward avoiding them when possible and controlling them when necessary.

There are two distinct types of risks for each of the categories that have been presented in Section 6.2: generic risks and product-specific risks. *Generic risks* are a potential threat to every software project. *Product-specific risks* can be identified only by those with a clear understanding of the technology, the people, and the environment that is specific to the project at hand. To identify product-specific risks, the project plan and the software statement of scope are examined and an answer to the following question is developed: "What special characteristics of this product may threaten our project plan?"

One method for identifying risks is to create a *risk item checklist*. The checklist can be used for risk identification and focuses on some subset of known and predictable risks in the following generic subcategories:

- Product size—risks associated with the overall size of the software to be built or modified.
- Business impact—risks associated with constraints imposed by management or the marketplace.
- Customer characteristics—risks associated with the sophistication of the customer and the developer's ability to communicate with the customer in a timely manner.
- Process definition—risks associated with the degree to which the software process has been defined and is followed by the development organization.
- *Development environment*—risks associated with the availability and quality of the tools to be used to build the product.
- *Technology to be built*—risks associated with the complexity of the system to be built and the "newness" of the technology that is packaged by the system.
- *Staff size and experience*—risks associated with the overall technical and project experience of the software engineers who will do the work.

The risk item checklist can be organized in different ways. Questions relevant to each of the topics can be answered for each software project. The answers to these questions allow the planner to estimate the impact of risk. A different risk item checklist format simply lists characteristics that are relevant to each generic subcategory. Finally, a set of "risk components and drivers" [AFC88] are listed along with their probability



Although generic risks are important to consider, usually the product-specific risks cause the most headaches. Be certain to spend the time to identify as many product-specific risks as possible.



Risk item checklist



Risk management is project management for adults."

**Tim Lister** 

of occurrence. Drivers for performance, support, cost, and schedule are discussed in answer to later questions.

A number of comprehensive checklists for software project risk have been proposed in the literature (e.g., [SEI93], [KAR96]). These provide useful insight into generic risks for software projects and should be used whenever risk analysis and management is instituted. However, a relatively short list of questions [KEI98] can be used to provide a preliminary indication of whether a project is "at risk."

#### 6.3.1 Assessing Overall Project Risk

The following questions have derived from risk data obtained by surveying experienced software project managers in different part of the world [KEI98]. The questions are ordered by their relative importance to the success of a project.

- **1.** Have top software and customer managers formally committed to support the project?
- **2.** Are end-users enthusiastically committed to the project and the system/product to be built?
- **3.** Are requirements fully understood by the software engineering team and their customers?
- **4.** Have customers been involved fully in the definition of requirements?
- **5.** Do end-users have realistic expectations?
- **6.** Is project scope stable?
- 7. Does the software engineering team have the right mix of skills?
- **8.** Are project requirements stable?
- **9.** Does the project team have experience with the technology to be implemented?
- 10. Is the number of people on the project team adequate to do the job?
- **11.** Do all customer/user constituencies agree on the importance of the project and on the requirements for the system/product to be built?

If any one of these questions is answered negatively, mitigation, monitoring, and management steps should be instituted without fail. The degree to which the project is at risk is directly proportional to the number of negative responses to these questions.

#### 6.3.2 Risk Components and Drivers

The U.S. Air Force [AFC88] has written a pamphlet that contains excellent guidelines for software risk identification and abatement. The Air Force approach requires that the project manager identify the risk drivers that affect software risk components—

Is the software project we're working on at serious risk?



Risk Radar is a risk management database that helps project managers identify, rank, and communicate project risks. It can be found at www.spmn.com/ rsktrkr.html performance, cost, support, and schedule. In the context of this discussion, the risk components are defined in the following manner:

- *Performance risk*—the degree of uncertainty that the product will meet its requirements and be fit for its intended use.
- Cost risk—the degree of uncertainty that the project budget will be maintained.
- *Support risk*—the degree of uncertainty that the resultant software will be easy to correct, adapt, and enhance.
- *Schedule risk*—the degree of uncertainty that the project schedule will be maintained and that the product will be delivered on time.

The impact of each risk driver on the risk component is divided into one of four impact categories—negligible, marginal, critical, or catastrophic. Referring to Figure 6.1 [BOE89],

Components						
		Performance	Support	Cost	Schedule	
Category						
Catastrophic	1	Failure to meet the requirement would result in mission failure		Failure results in increased costs and schedule delays with expected values in excess of \$500K		
	2	Significant degradation to nonachievement of technical performance	Nonresponsive or unsupportable software	Significant financial shortages, budget overrun likely	Unachievable IOC	
Critical	1	Failure to meet the requirement would degrade system performance to a point where mission success is questionable		Failure results in operational delays and/or increased costs with expected value of \$100K to \$500K		
	2	Some reduction in technical performance	Minor delays in software modifications	Some shortage of financial resources, possible overruns	Possible slippage in IOC	
Marginal 2	1	Failure to meet the requirement would result in degradation of secondary mission		Costs, impacts, and/or recoverable schedule slips with expected value of \$1K to \$100K		
	2	Minimal to small reduction in technical performance	Responsive software support	Sufficient financial resources	Realistic, achievable schedule	
Negligible	1	Failure to meet the requirement would create inconvenience or nonoperational impact		Error results in minor cost and/or schedule impact with expected value of less than \$1K		
	2	No reduction in technical performance	Easily supportable software	Possible budget underrun	Early achievable IOC	

Note: (1) The potential consequence of undetected software errors or faults.

(2) The potential consequence if the desired outcome is not achieved.

FIGURE 6.1 Impact assessment [BOE89]

a characterization of the potential consequences of errors (rows labeled 1) or a failure to achieve a desired outcome (rows labeled 2) are described. The impact category is chosen based on the characterization that best fits the description in the table.

#### 6.4 RISK PROJECTION

Risk projection, also called risk estimation, attempts to rate each risk in two ways—the likelihood or probability that the risk is real and the consequences of the problems associated with the risk, should it occur. The project planner, along with other managers and technical staff, performs four risk projection activities: (1) establish a scale that reflects the perceived likelihood of a risk, (2) delineate the consequences of the risk, (3) estimate the impact of the risk on the project and the product, and (4) note the overall accuracy of the risk projection so that there will be no misunderstandings.

#### 6.4.1 Developing a Risk Table

A risk table provides a project manager with a simple technique for risk projection.<sup>2</sup> A sample risk table is illustrated in Figure 6.2.

Risks	Category	Probability	Impact	RMMM
Size estimate may be significantly low	PS	60%	2	
Larger number of users than planned	PS	30%	3	
Less reuse than planned	PS	70%		
End-users resist system	BU	40%	2 3	
Delivery deadline will be tightened	BU	50%	2	
Funding will be lost	CU	40%	1	
Customer will change requirements	PS	80%	2	
Technology will not meet expectations	TE	30%	1	
Lack of training on tools	DE	80%	3	
Staff inexperienced	ST	30%	2	
Staff turnover will be high	ST	60%	2	
•				
•				
•				

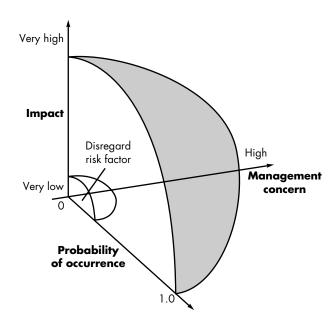
Impact values:

- 1—catastrophic
- 2—critical
- 3—marginal
- 4-negligible

FIGURE 6.2 Sample risk table prior to sorting

<sup>2</sup> The risk table should be implemented as a spreadsheet model. This enables easy manipulation and sorting of the entries.

FIGURE 6.3
Risk and
management
concern





Think hard about the software you're about to build and ask yourself, "What can go wrong?" Create your own list and ask other members of the software team to do the same.



The risk table is sorted by probability and impact to rank risks.

A project team begins by listing all risks (no matter how remote) in the first column of the table. This can be accomplished with the help of the risk item checklists referenced in Section 6.3. Each risk is categorized in the second column (e.g., PS implies a project size risk, BU implies a business risk). The probability of occurrence of each risk is entered in the next column of the table. The probability value for each risk can be estimated by team members individually. Individual team members are polled in round-robin fashion until their assessment of risk probability begins to converge.

Next, the impact of each risk is assessed. Each risk component is assessed using the characterization presented in Figure 6.1, and an impact category is determined. The categories for each of the four risk components—performance, support, cost, and schedule—are averaged<sup>3</sup> to determine an overall impact value.

Once the first four columns of the risk table have been completed, the table is sorted by probability and by impact. High-probability, high-impact risks percolate to the top of the table, and low-probability risks drop to the bottom. This accomplishes first-order risk prioritization.

The project manager studies the resultant sorted table and defines a cutoff line. The *cutoff line* (drawn horizontally at some point in the table) implies that only risks that lie above the line will be given further attention. Risks that fall below the line are re-evaluated to accomplish second-order prioritization. Referring to Figure 6.3, risk impact and probability have a distinct influence on management concern. A risk fac-

<sup>3</sup> A weighted average can be used if one risk component has more significance for the project.

tor that has a high impact but a very low probability of occurrence should not absorb a significant amount of management time. However, high-impact risks with moderate to high probability and low-impact risks with high probability should be carried forward into the risk analysis steps that follow.

All risks that lie above the cutoff line must be managed. The column labeled RMMM contains a pointer into a Risk Mitigation, Monitoring and Management Plan or alternatively, a collection of risk information sheets developed for all risks that lie above the cutoff. The RMMM plan and risk information sheets are discussed in Sections 6.5 and 6.6.

Risk probability can be determined by making individual estimates and then developing a single consensus value. Although that approach is workable, more sophisticated techniques for determining risk probability have been developed [AFC88]. Risk drivers can be assessed on a qualitative probability scale that has the following values: impossible, improbable, probable, and frequent. Mathematical probability can then be associated with each qualitative value (e.g., a probability of 0.7 to 1.0 implies a highly probable risk).

#### 6.4.2 Assessing Risk Impact

Three factors affect the consequences that are likely if a risk does occur: its nature, its scope, and its timing. The *nature* of the risk indicates the problems that are likely if it occurs. For example, a poorly defined external interface to customer hardware (a technical risk) will preclude early design and testing and will likely lead to system integration problems late in a project. The scope of a risk combines the severity (just how serious is it?) with its overall distribution (how much of the project will be affected or how many customers are harmed?). Finally, the timing of a risk considers when and for how long the impact will be felt. In most cases, a project manager might want the "bad news" to occur as soon as possible, but in some cases, the longer the delay, the better.

Returning once more to the risk analysis approach proposed by the U.S. Air Force [AFC88], the following steps are recommended to determine the overall consequences of a risk:

- 1. Determine the average probability of occurrence value for each risk component.
- 2. Using Figure 6.1, determine the impact for each component based on the criteria shown.
- 3. Complete the risk table and analyze the results as described in the preceding sections.

The overall risk exposure, RE, is determined using the following relationship [HAL98]:

 $RE = P \times C$ 

uote:

Failure to prepare is preparing to fail." Ben Franklin



where *P* is the probability of occurrence for a risk, and *C* is the the cost to the project should the risk occur.

For example, assume that the software team defines a project risk in the following manner:

**Risk identification.** Only 70 percent of the software components scheduled for reuse will, in fact, be integrated into the application. The remaining functionality will have to be custom developed.

Risk probability. 80% (likely).

**Risk impact.** 60 reusable software components were planned. If only 70 percent can be used, 18 components would have to be developed from scratch (in addition to other custom software that has been scheduled for development). Since the average component is 100 LOC and local data indicate that the software engineering cost for each LOC is \$14.00, the overall cost (impact) to develop the components would be  $18 \times 100 \times 14 = $25,200$ .

**Risk exposure.**  $RE = 0.80 \times 25,200 \sim $20,200.$ 

Risk exposure can be computed for each risk in the risk table, once an estimate of the cost of the risk is made. The total risk exposure for all risks (above the cutoff in the risk table) can provide a means for adjusting the final cost estimate for a project. It can also be used to predict the probable increase in staff resources required at various points during the project schedule.

The risk projection and analysis techniques described in Sections 6.4.1 and 6.4.2 are applied iteratively as the software project proceeds. The project team should revisit the risk table at regular intervals, re-evaluating each risk to determine when new circumstances cause its probability and impact to change. As a consequence of this activity, it may be necessary to add new risks to the table, remove some risks that are no longer relevant, and change the relative positions of still others.

#### 6.4.3 Risk Assessment

At this point in the risk management process, we have established a set of triplets of the form [CHA89]:

$$[r_i, l_i, x_i]$$

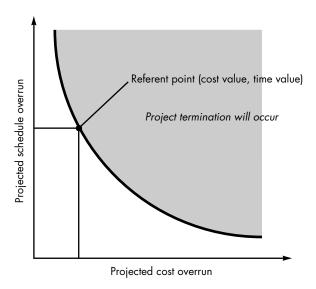
where  $r_i$  is risk,  $l_i$  is the likelihood (probability) of the risk, and  $x_i$  is the impact of the risk. During risk assessment, we further examine the accuracy of the estimates that were made during risk projection, attempt to rank the risks that have been uncovered, and begin thinking about ways to control and/or avert risks that are likely to occur.

For assessment to be useful, a *risk referent level* [CHA89] must be defined. For most software projects, the risk components discussed earlier—performance, cost, support, and schedule—also represent risk referent levels. That is, there is a level for per-



Compare RE for all risks to the cost estimate for the project. If RE is greater than 50 percent of project cost, the viability of the project must be evaluated.

FIGURE 6.4 Risk referent level





The risk referent level establishes your tolerance for pain. Once risk exposure exceeds the referent level, the project may be terminated.

formance degradation, cost overrun, support difficulty, or schedule slippage (or any combination of the four) that will cause the project to be terminated. If a combination of risks create problems that cause one or more of these referent levels to be exceeded, work will stop. In the context of software risk analysis, a risk referent level has a single point, called the *referent point* or *break point*, at which the decision to proceed with the project or terminate it (problems are just too great) are equally weighted. Figure 6.4 represents this situation graphically.

In reality, the referent level can rarely be represented as a smooth line on a graph. In most cases it is a region in which there are areas of uncertainty; that is, attempting to predict a management decision based on the combination of referent values is often impossible. Therefore, during risk assessment, we perform the following steps:

- 1. Define the risk referent levels for the project.
- **2.** Attempt to develop a relationship between each  $(r_i, l_i, x_i)$  and each of the referent levels.
- **3.** Predict the set of referent points that define a region of termination, bounded by a curve or areas of uncertainty.
- **4.** Try to predict how compound combinations of risks will affect a referent level.

A detailed discussion of risk referent level is best left to books that are dedicated to risk analysis (e.g., [CHA89], [ROW88]).

#### 6.5 RISK REFINEMENT

During early stages of project planning, a risk may be stated quite generally. As time passes and more is learned about the project and the risk, it may be possible to refine the risk into a set of more detailed risks, each somewhat easier to mitigate, monitor, and manage.

What is a good way to describe a risk?

One way to do this is to represent the risk in *condition-transition-consequence* (CTC) format [GLU94]. That is, the risk is stated in the following form:

Given that <condition> then there is concern that (possibly) <consequence>.

Using the CTC format for the reuse risk noted in Section 6.4.2, we can write:

Given that all reusable software components must conform to specific design standards and that some do not conform, then there is concern that (possibly) only 70 percent of the planned reusable modules may actually be integrated into the as-built system, resulting in the need to custom engineer the remaining 30 percent of components.

This general condition can be refined in the following manner:

**Subcondition 1.** Certain reusable components were developed by a third party with no knowledge of internal design standards.

**Subcondition 2.** The design standard for component interfaces has not been solidified and may not conform to certain existing reusable components.

**Subcondition 3.** Certain reusable components have been implemented in a language that is not supported on the target environment.

The consequences associated with these refined subconditions remains the same (i.e., 30 percent of software components must be customer engineered), but the refinement helps to isolate the underlying risks and might lead to easier analysis and response.

#### 6.6 RISK MITIGATION, MONITORING, AND MANAGEMENT

All of the risk analysis activities presented to this point have a single goal—to assist the project team in developing a strategy for dealing with risk. An effective strategy must consider three issues:

- risk avoidance
- risk monitoring
- risk management and contingency planning

If a software team adopts a proactive approach to risk, avoidance is always the best strategy. This is achieved by developing a plan for *risk mitigation*. For example, assume that high staff turnover is noted as a project risk,  $r_1$ . Based on past history and man-



"If I take so many precautions, it is because I leave nothing to chance."

Napolean



An excellent FAQ on risk management can be obtained at www.sei.cmu.edu/ organization/ programs/sepm/ risk/risk.faq.html agement intuition, the likelihood,  $l_I$ , of high turnover is estimated to be 0.70 (70 percent, rather high) and the impact,  $x_I$ , is projected at level 2. That is, high turnover will have a critical impact on project cost and schedule.

To mitigate this risk, project management must develop a strategy for reducing turnover. Among the possible steps to be taken are

- Meet with current staff to determine causes for turnover (e.g., poor working conditions, low pay, competitive job market).
- Mitigate those causes that are under our control before the project starts.
- Once the project commences, assume turnover will occur and develop techniques to ensure continuity when people leave.
- Organize project teams so that information about each development activity is widely dispersed.
- Define documentation standards and establish mechanisms to be sure that documents are developed in a timely manner.
- Conduct peer reviews of all work (so that more than one person is "up to speed").
- Assign a backup staff member for every critical technologist.

As the project proceeds, risk monitoring activities commence. The project manager monitors factors that may provide an indication of whether the risk is becoming more or less likely. In the case of high staff turnover, the following factors can be monitored:

General attitude of team members based on project pressures.

- The degree to which the team has jelled.
- Interpersonal relationships among team members.
- Potential problems with compensation and benefits.
- The availability of jobs within the company and outside it.

In addition to monitoring these factors, the project manager should monitor the effectiveness of risk mitigation steps. For example, a risk mitigation step noted here called for the definition of documentation standards and mechanisms to be sure that documents are developed in a timely manner. This is one mechanism for ensuring continuity, should a critical individual leave the project. The project manager should monitor documents carefully to ensure that each can stand on its own and that each imparts information that would be necessary if a newcomer were forced to join the software team somewhere in the middle of the project.

Risk management and contingency planning assumes that mitigation efforts have failed and that the risk has become a reality. Continuing the example, the project is



'We are ready for an unforseen event that may or may not occur."

**Dan Quayle** 

well underway and a number of people announce that they will be leaving. If the mitigation strategy has been followed, backup is available, information is documented, and knowledge has been dispersed across the team. In addition, the project manager may temporarily refocus resources (and readjust the project schedule) to those functions that are fully staffed, enabling newcomers who must be added to the team to "get up to speed." Those individuals who are leaving are asked to stop all work and spend their last weeks in "knowledge transfer mode." This might include video-based knowledge capture, the development of "commentary documents," and/or meeting with other team members who will remain on the project.



If RE for a specific risk is less than the cost of risk mitigation, don't try to mitigate the risk but continue to monitor it. It is important to note that RMMM steps incur additional project cost. For example, spending the time to "backup" every critical technologist costs money. Part of risk management, therefore, is to evaluate when the benefits accrued by the RMMM steps are outweighed by the costs associated with implementing them. In essence, the project planner performs a classic cost/benefit analysis. If risk aversion steps for high turnover will increase both project cost and duration by an estimated 15 percent, but the predominant cost factor is "backup," management may decide not to implement this step. On the other hand, if the risk aversion steps are projected to increase costs by 5 percent and duration by only 3 percent management will likely put all into place.

For a large project, 30 or 40 risks may identified. If between three and seven risk management steps are identified for each, risk management may become a project in itself! For this reason, we adapt the Pareto 80–20 rule to software risk. Experience indicates that 80 percent of the overall project risk (i.e., 80 percent of the potential for project failure) can be accounted for by only 20 percent of the identified risks. The work performed during earlier risk analysis steps will help the planner to determine which of the risks reside in that 20 percent (e.g., risks that lead to the highest risk exposure). For this reason, some of the risks identified, assessed, and projected may not make it into the RMMM plan—they don't fall into the critical 20 percent (the risks with highest project priority).

#### 6.7 SAFETY RISKS AND HAZARDS

Risk is not limited to the software project itself. Risks can occur after the software has been successfully developed and delivered to the customer. These risks are typically associated with the consequences of software failure in the field.

In the early days of computing, there was reluctance to use computers (and software) to control safety critical processes such as nuclear reactors, aircraft flight control, weapons systems, and large-scale industrial processes. Although the probability of failure of a well-engineered system was small, an undetected fault in a computer-based control or monitoring system could result in enormous economic damage or, worse, significant human injury or loss of life. But the cost and functional benefits of



A voluminous database containing all entries from the ACM's Forum on Risks to the Public can be found at

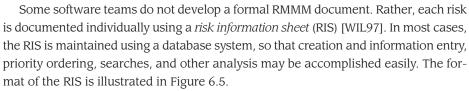
catless.ncl.ac.uk/ Risks/search.html computer-based control and monitoring far outweigh the risk. Today, computer hardware and software are used regularly to control safety critical systems.

When software is used as part of a control system, complexity can increase by an order of magnitude or more. Subtle design faults induced by human error—something that can be uncovered and eliminated in hardware-based conventional control—become much more difficult to uncover when software is used.

Software safety and hazard analysis [LEV95] are software quality assurance activities (Chapter 8) that focus on the identification and assessment of potential hazards that may affect software negatively and cause an entire system to fail. If hazards can be identified early in the software engineering process, software design features can be specified that will either eliminate or control potential hazards.

#### 6.8 THE RMMM PLAN

A risk management strategy can be included in the software project plan or the risk management steps can be organized into a separate *Risk Mitigation, Monitoring and Management Plan*. The RMMM plan documents all work performed as part of risk analysis and is used by the project manager as part of the overall project plan.



Once RMMM has been documented and the project has begun, risk mitigation and monitoring steps commence. As we have already discussed, risk mitigation is a problem avoidance activity. Risk monitoring is a project tracking activity with three primary objectives: (1) to assess whether predicted risks do, in fact, occur; (2) to ensure that risk aversion steps defined for the risk are being properly applied; and (3) to collect information that can be used for future risk analysis. In many cases, the problems that occur during a project can be traced to more than one risk. Another job of risk monitoring is to attempt to allocate *origin* (what risk(s) caused which problems throughout the project).

#### 6.9 SUMMARY

Whenever a lot is riding on a software project, common sense dictates risk analysis. And yet, most software project managers do it informally and superficially, if they do it at all. The time spent identifying, analyzing, and managing risk pays itself back in many ways: less upheaval during the project, a greater ability to track and control a project, and the confidence that comes with planning for problems before they occur.



#### FIGURE 6.5

Risk information sheet [WIL97]

Risk information sheet								
Risk ID: P02-4-32	Date: 5/9/02	Prob: 80%	Impact: high					

#### **Description:**

Only 70 percent of the software components scheduled for reuse will, in fact, be integrated into the application. The remaining functionality will have to be custom developed.

#### Refinement/context:

Subcondition 1: Certain reusable components were developed by a third party with no knowledge of internal design standards.

Subcondition 2: The design standard for component interfaces has not been solidified and may not conform to certain existing reusable components. Subcondition 3: Certain reusable components have been implemented in a language that is not supported on the target environment.

#### Mitigation/monitoring:

- 1. Contact third party to determine conformance with design standards.
- 2. Press for interface standards completion; consider component structure when deciding on interface protocol.
- 3. Check to determine number of components in subcondition 3 category; check to determine if language support can be acquired.

#### Management/contingency plan/trigger:

RE computed to be \$20,200. Allocate this amount within project contingency cost. Develop revised schedule assuming that 18 additional components will have to be custom built; allocate staff accordingly.

Trigger: Mitigation steps unproductive as of 7/1/02

#### **Current status:**

5/12/02: Mitigation steps initiated.

Originator: D. Gagne Assigned: B. Laster

Risk analysis can absorb a significant amount of project planning effort. Identification, projection, assessment, management, and monitoring all take time. But the effort is worth it. To quote Sun Tzu, a Chinese general who lived 2500 years ago, "If you know the enemy and know yourself, you need not fear the result of a hundred battles." For the software project manager, the enemy is risk.

#### REFERENCES

[AFC88] *Software Risk Abatement,* AFCS/AFLC Pamphlet 800-45, U.S. Air Force, September 30, 1988.

[BOE89] Boehm, B.W., *Software Risk Management,* IEEE Computer Society Press, 1989.

[CHA89] Charette, R.N., Software Engineering Risk Analysis and Management, McGraw-Hill/Intertext, 1989.

[CHA92] Charette, R.N., "Building Bridges over Intelligent Rivers," *American Programmer*, vol. 5, no. 7, September, 1992, pp. 2–9.

[DRU75] Drucker, P., Management, W. H. Heinemann, 1975.

[GIL88] Gilb, T., Principles of Software Engineering Management, Addison-Wesley, 1988.

[GLU94] Gluch, D.P., "A Construct for Describing Software Development Risks," CMU/SEI-94-TR-14, Software Engineering Institute, 1994.

[HAL98] Hall, E.M., *Managing Risk: Methods for Software Systems Development,* Addison-Wesley, 1998.

[HIG95] Higuera, R.P., "Team Risk Management," *CrossTalk,* U.S. Dept. of Defense, January 1995, p. 2–4.

[KAR96] Karolak, D.W., *Software Engineering Risk Management,* IEEE Computer Society Press, 1996.

[KEI98] Keil, M., et al., "A Framework for Identifying Software Project Risks," *CACM*, vol. 41, no. 11, November 1998, pp. 76–83.

[LEV95] Leveson, N.G., Safeware: System Safety and Computers, Addison-Wesley, 1995.

[ROW88] Rowe, W.D., An Anatomy of Risk, Robert E. Krieger Publishing Co., 1988.

[SEI93] "Taxonomy-Based Risk Identification," Software Engineering Institute, CMU/SEI-93-TR-6, 1993.

[THO92] Thomsett, R., "The Indiana Jones School of Risk Management," *American Programmer*, vol. 5, no. 7, September 1992, pp. 10–18.

[WIL97] Williams, R.C, J.A. Walker, and A.J. Dorofee, "Putting Risk Management into Practice," *IEEE Software*, May 1997, pp. 75–81.

#### PROBLEMS AND POINTS TO PONDER

- **6.1.** Provide five examples from other fields that illustrate the problems associated with a reactive risk strategy.
- **6.2.** Describe the difference between "known risks" and "predictable risks."
- **6.3.** Add three additional questions or topics to each of the risk item checklists presented at the SEPA Web site.
- **6.4.** You've been asked to build software to support a low-cost video editing system. The system accepts videotape as input, stores the video on disk, and then allows the user to do a wide range of edits to the digitized video. The result can then be output to tape. Do a small amount of research on systems of this type and then make a list of technology risks that you would face as you begin a project of this type.
- **6.5.** You're the project manager for a major software company. You've been asked to lead a team that's developing "next generation" word-processing software (see Section 3.4.2 for a brief description). Create a risk table for the project.

- **6.6.** Describe the difference between risk components and risk drivers.
- **6.7.** Develop a risk mitigation strategy and specific risk mitigation activities for three of the risks noted in Figure 6.2.
- **6.8.** Develop a risk monitoring strategy and specific risk monitoring activities for three of the risks noted in Figure 6.2. Be sure to identify the factors that you'll be monitoring to determine whether the risk is becoming more or less likely.
- **6.9.** Develop a risk management strategy and specific risk management activities for three of the risks noted in Figure 6.2.
- **6.10.** Attempt to refine three of the risks noted in Figure 6.2 and then create risk information sheets for each.
- **6.11.** Represent three of the risks noted in Figure 6.2 using a CTC format.
- **6.12.** Recompute the risk exposure discussed in Section 6.4.2 when cost/LOC is \$16 and the probability is 60 percent.
- **6.13.** Can you think of a situation in which a high-probability, high-impact risk would not be considered as part of your RMMM plan?
- **6.14.** Referring the the risk referent shown on Figure 6.4, would the curve always have the symmetric arc shown or would there be situations in which the curve would be more distorted. If so, suggest a scenario in which this might happen.
- **6.15.** Do some research on software safety issues and write a brief paper on the subject. Do a Web search to get current information.
- **6.16.** Describe five software application areas in which software safety and hazard analysis would be a major concern.

#### FURTHER READINGS AND INFORMATION SOURCES

The software risk management literature has expanded significantly in recent years. Hall [HAL98] presents one of the more thorough treatments of the subject. Karolak [KAR96] has written a guidebook that introduces an easy-to-use risk analysis model with worthwhile checklists and questionnaires. A useful snapshot of risk assessment has been written by Grey (*Practical Risk Assessment for Project Management,* Wiley, 1995). His abbreviated treatment provides a good introduction to the subject. Additional books worth examining include

Chapman, C.B. and S. Ward, *Project Risk Management: Processes, Techniques and Insights,* Wiley, 1997.

Schuyler, J.R., Decision Analysis in Projects, Project Management Institute Publications, 1997.

Wideman, R.M. (editor), *Project & Program Risk Management: A Guide to Managing Project Risks and Opportunities*, Project Management Institute Publications, 1998.

Capers Jones (Assessment and Control of Software Risks, Prentice-Hall, 1994) presents a detailed discussion of software risks that includes data collected from hundreds of software projects. Jones defines 60 risk factors that can affect the outcome of software projects. Boehm [BOE89] suggests excellent questionnaire and checklist formats that can prove invaluable in identifying risk. Charette [CHA89] presents a detailed treatment of the mechanics of risk analysis, calling on probability theory and statistical techniques to analyze risks. In a companion volume, Charette (Application Strategies for Risk Analysis, McGraw-Hill, 1990) discusses risk in the context of both system and software engineering and suggests pragmatic strategies for risk management. Gilb (Principles of Software Engineering Management, Addison-Wesley, 1988) presents a set of "principles" (which are often amusing and sometimes profound) that can serve as a worthwhile guide for risk management.

The March 1995 issue of *American Programmer*, the May 1997 issue of *IEEE Software*, and the June 1998 issue of the *Cutter IT Journal* all are dedicated to risk management.

The Software Engineering Institute has published many detailed reports and guidebooks on risk analysis and management. The Air Force Systems Command pamphlet AFSCP 800-45 [AFC88] describes risk identification and reduction techniques. Every issue of the *ACM Software Engineering Notes* has a section entitled "Risks to the Public" (editor, P.G. Neumann). If you want the latest and best software horror stories, this is the place to go.

A wide variety of information sources on risk analysis and management is available on the Internet. An up-to-date list of World Wide Web references that are relevant to risk can be found at the SEPA Web site:

http://www.mhhe.com/engcs/compsci/pressman/resources/risk.mhtml