**6-1 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:

* **Uncertainty**: the risk may or may not happen; that is, there are no 100% probable risks.
* **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.

 **Project 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 risk the project or the product. Candidates for the top ﬁve business risks are

(1) Building an 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:

 ***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.

**6-2 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 6.1: 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" are listed along with their probability 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 can be used to provide a preliminary indication of whether a project is “at risk.”

**6-2-1 Assessing Overall Risk**

 The following questions have derived from risk data obtained by surveying experienced software project managers in different part of the world. 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.

**6-2-2 Risk Components and Drivers**

 The U.S. Air Force 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 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.



Figure 6-1 Impact Assessment

**6-3 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.

* **Developing a Risk Table**

 A risk table provides a project manager with a simple technique for risk projection. A sample risk table is illustrated in Figure 6.2.



Figure 6-2 Sample Risk Table Prior to Sorting

Impact values:

1- Catastrophic

2-critical

3-marginal

4-negligible

 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 check-lists 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 averaged3 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 7.3, risk impact and probability have a distinct influence on management concern.



Figure (6.3) Risk and management concern

All risks that lie above the cutoff line must be managed. The column labeled RMMM in figure 7.2 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.

**6.4 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?).
* The *timing* of a risk considers when and for how long the impact will be felt.

- How do we assess the 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:

RE = P x C

Where P is the probability of occurrence for a risk, and C is 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 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 (custom software). 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 x 100 x 14 = $25,200.

Risk exposure: RE = 0.80 x 25,200 ~ $20,200.

**7.5 Risk Assessment**

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

[*ri, li, xi*]

Where *ri* is risk, *li* is the likelihood (probability) of the risk, and *xi* 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 risks that are likely to occur.

**7.6 The Risk Mitigation, Monitoring and Management(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 are used by the project manager as part of the overall project plan.

Some software teams do not develop a formal RMMM document. Rather, each risk is documented individually using a *risk information sheet* (RIS). In most cases, the RIS is maintained using a database system, so that creation and information entry, priority ordering, searches, and other analysis may be accomplished easily. The format of the RIS is illustrated in Figure 7.4.



**Figure (6.4) Risk information sheet**