SOFTWARE RISK MANAGEMENT

Linda Westfall
The Westfall Team
westfall@idt.net
PMB 383, 3000 Custer Road, Suite 270
Plano, TX 75075
972-867-1172 (voice)
972-943-1484 (fax)

SUMMARY

This paper reviews the basic concepts, terminology, and techniques of Software Risk Management. It teaches readers how to identify and analyze software risks on their projects. Readers then learn techniques for planning and acting to mitigate risks so that the overall impact of those risks on their projects is minimized.

KEY WORDS

Risk Analysis, Risk Identification, Software Project Management

DEFINING SOFTWARE RISK MANAGEMENT

There are many risks involved in creating high quality software on time and within budget. However, in order for it to be worthwhile to take these risks, they must be compensated for by a perceived reward. The greater the risk, the greater the reward must be to make it worthwhile to take the chance. In software development, the possibility of reward is high, but so is the potential for disaster. The need for software risk management is illustrated in Gilb's risk principle. "If you don't actively attack the risks, they will actively attack you" [Gilb-88]. In order to successfully manage a software project and reap our rewards, we must learn to identify, analyze, and control these risks. This paper focuses on the basic concepts, processes, and techniques of software risk management.

There are basic risks that are generic to almost all software projects. Although there is a basic component of risk management inherent in good project management, risk management differs from project management in the following ways:

Project Management	Risk Management
Designed to address general or generic risks	Designed to focus on risks unique to each project
Looks at the big picture and plans for details	Looks at potential problems and plans for contingencies
Plans what should happen and looks for ways to make it happen	Evaluates what could happen and looks for ways to minimize the damage
Plans for success	Plans to manage and mitigate potential causes of failure

Within risk management the "emphasis is shifted from crisis management to anticipatory management" [Down-94].

Boehm defines four major reasons for implementing software risk management [Boehm-89]:

1. Avoiding software project disasters, including run away budgets and schedules, defect-ridden software products, and operational failures.
2. Avoiding rework caused by erroneous, missing, or ambiguous requirements, design or code, which typically consumes 40-50% of the total cost of software development.
3. Avoiding overkill with detection and prevention techniques in areas of minimal or no risk.
4. Stimulating a win-win software solution where the customer receives the product they need and the vendor makes the profits they expect.

Copyright © 2001 The Westfall Team. All Rights Reserved

DEFINING RISK

So, what are risks? Risks are simply potential problems. For example, every time we cross the street, we run the risk of being hit by a car. The risk does not start until we make the commitment, until we step in the street. It ends when the problem occurs (the car hits us) or the possibility of risk is eliminated (we safely step onto the sidewalk of the other side of the street).

A software project may encounter various types of risks:

- **Technical risks** include problems with languages, project size, project functionality, platforms, methods, standards, or processes. These risks may result from excessive constraints, lack of experience, poorly defined parameters, or dependencies on organizations outside the direct control of the project team.
- **Management risks** include lack of planning, lack of management experience and training, communications problems, organizational issues, lack of authority, and control problems.
- **Financial risks** include cash flow, capital and budgetary issues, and return on investment constraints.
- **Contractual and legal risks** include changing requirements, market-driven schedules, health & safety issues, government regulation, and product warranty issues.
- **Personnel risks** include staffing lags, experience and training problems, ethical and moral issues, staff conflicts, and productivity issues.
- **Other resource risks** include unavailability or late delivery of equipment & supplies, inadequate tools, inadequate facilities, distributed locations, unavailability of comp uter resources, and slow response times.

RISK MANAGEMENT PROCESS

Figure 1 illustrates the risk management process. This process starts with the identification of a list of potential risks. Each of these risks is then analyzed and prioritized. A risk management plan is created that identifies containment actions that will reduce the probability of the risk occurring and/or reduce the impact if the risk turns into a problem. The plan also includes contingency actions that will be taken if the risk turns into a problem and the associated triggers (indicators that the risk is turning into a problem). The containment part of the plan is then implemented and actions are taken. The tracking step involves monitoring the status of known risks as well as the results of risk reduction actions. If a trigger indicates the onset of a problem, the corresponding contingency plans are implemented. As new status and information are obtained, the risk management plans are updated accordingly. Tracking may also result in the addition of newly identified risks or in the closure of known risks.

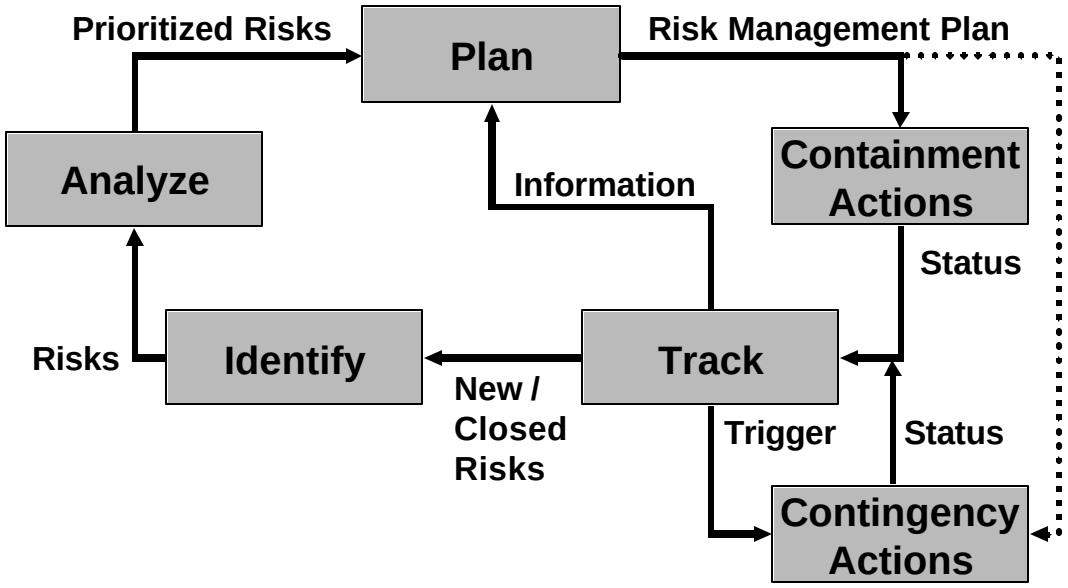

Figure 1 - Risk Management Process

Copyright © 2001 The Westfall Team. All Rights Reserved

The risk management process is an on-going part of managing the software development process. It is designed to be a continuous feedback loop where additional information and risk status are utilized to refine the project's risk list and risk management plans.

Let's use the crossing the street analogy to examine the risk management process. First we identify the risk: we want to cross the street and know there is a possibility of traffic. We analyze the risk. What is the probability of being hit by the car? How much is it going to hurt if we are hit? How important is it that we cross this street at this time? We look both ways, we see the on-coming car, and we judge its rate of speed. We form a plan to reduce the risk and decide to wait until the car has passed. We implement the plan and wait. We track the situation by watching the car and we see it pull into a driveway. We change our plan and proceed across the street. We step onto the curb across the street and stop thinking about crossing the street (i.e., we close the risk).

RISK IDENTIFICATION

During the first step in the software risk management process, risks are identified and added to the list of known risks. The output of this step is a list of project-specific risks that have the potential of compromising the project's success. There are many techniques for identifying risks, including interviewing, reporting, decomposition, assumption analysis, critical path analysis, and utilization of risk taxonomies.

Interviewing/Brainstorming: One technique for identifying risks is interviewing or brainstorming with project personnel, customers, and vendors. Open-ended questions such as the following can help identify potential areas of risk.

- What new or improved technologies does this project implement?
- What interfaces issues still need to be defined?
- What requirements exist that we aren't sure how to implement?
- What concerns do we have about our ability to meet the required quality and performance levels?

Voluntary Reporting: Another risk identification technique is voluntary reporting, where any individual who identifies a risk is encouraged and rewarded for bringing that risk to management's attention. This requires the complete elimination of the "shoot the messenger" syndrome. It avoids the temptation to assign risk reduction actions to the person who identified the risk. Risks can also be identified through required reporting mechanisms such as status reports or project reviews.

Decomposition: As the product is being decomposed during the requirements and design phases, another opportunity exists for risk identifications. Every TBD ("To Be Done/Determined") is a potential risk. As Ould states, "The most important thing about planning is writing down what you ***don't know***, because what you don't know is what you must find out" [Ould-90]. Decomposition in the form of work breakdown structures during project planning can also help identify areas of uncertainty that may need to be recorded as risks.

Assumption Analysis: Process and product assumptions must be analyzed. For example, we might assume the hardware would be available by the system test date or three additional experienced C++ programmers will be hired by the time coding starts. If these assumptions prove to be false, we could have major problems.

Critical Path Analysis: As we perform critical path analysis for our project plan, we must remain on the alert to identify risks. Any possibility of schedule slippage on the critical path must be considered a risk because it directly impacts our ability to meet schedule.

Risk Taxonomies: Risk taxonomies are lists of problems that have occurred on other projects and can be used as checklists to help ensure all potential risks have been considered. An example of a risk taxonomy can be found in the Software Engineering Institute's Taxonomy-Based Risk Identification report that covers 13 major risk areas with about 200 questions [SEI-93].

Copyright © 2001 The Westfall Team. All Rights Reserved

RISK ANALYSIS

During the risk analysis step, each risk is assessed to determine:

- Likelihood: the probability that the risk will result in a loss
- Impact: the size or cost of that loss if the risk turns into a problem
- Timeframe: when the risk needs to be addressed (i.e., risk associated with activities in the near future would have a higher priority then similar risks in later activities)

Additionally, the interrelationships between risks are assessed to determine if compounding risk conditions magnify losses.

The following is an example of risk analysis. During our analysis, we determine that there is a 30% probability the Test Bed will be available one week later than scheduled and a 10% probability it will be a month late. If the Test Bed is one week late, the testers can use their time productively by using the simulators to test other aspects of the software (loss = $0). The simulator can be utilized for up to two weeks. However, if the Test Bed delivery is one month late, there are not enough productive activities to balance the loss. Losses include unproductive testers for two weeks, overtime later, morale problems, and delays in finding defects for a total estimated loss of $100,000. In addition to the dollar loss, the testing is on the critical path and not all of the lost testing time can be made up in overtime (loss estimated at two week schedule slippage).

Boehm defines the Risk Exposure equation to help quantitatively establish risk priorities [Boehm-89]. Risk Exposure measures the impact of a risk in terms of the expected value of the loss. Risk Exposure (RE) is defined as the probability of an undesired outcome times the expected loss if that outcome occurs.

RE = Probability(UO) * Loss (UO), where UO = Unexpected outcome

Given the example above, the Risk Exposure is 10% x $100,000 = $10,000 and 10% x 2 calendar week = 0.2 calendar week. Comparing the Risk Exposure measurement for various risks can help identify those risks with the greatest probable negative impact to the project or product and thus help establish which risks are candidates for further action.

The list of risks is then prioritized based on the results of our risk analysis. Since resource limitations rarely allow the consideration of all risks, the prioritized list of risks is used to identify risks requiring additional planning and action. Other risks are documented and tracked for possible future consideration. Based on changing conditions, additional information, the identification of new risks, or the closure of existing risks, the list of risks requiring additional planning and action may require periodic updates.

RISK MANAGEMENT PLANNING

Taking the prioritized risk list as input, plans are developed for the risks chosen for action. Figure 2 illustrates the specific questions that can be asked to help focus on the type of planning required. We will use the following two risks to illustrate the types of actions that might be taken using each risk handling technique:

- The subcontractor may not deliver the software at the required reliability level and as a result the reliability of the total system may not meet performance specifications.
- The interface with the new

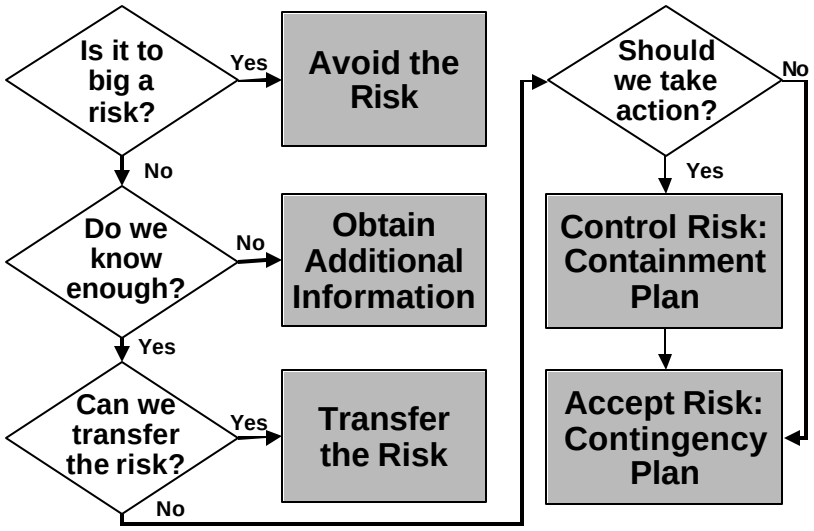

Figure 2 - Techniques for Handling Risks

Copyright © 2001 The Westfall Team. All Rights Reserved

control device is not defined and as a result its driver may take more time to implement then scheduled.

Is it too big a risk? If the risk is too big for us to be willing to accept, we can avoid the risk by changing our project strategies and tactics to choose a less risky alternate or we may decide not to do the project at all. For example, if our project has tight schedule constraints and includes state of the art technology, we may decide to wait until a future project to implement our newly purchased CASE tools.

Things to remember about avoiding risks include:

- Avoiding risks may also mean avoiding opportunities
- Not all risks can be avoided
- Avoiding a risk in one part of the project may create risks in other parts of the project

Risk	Avoid the Risk
The subcontractor may not deliver the software at the required reliability level and as a result the reliability of the total system may not meet performance specifications.	Develop all software in-house. Switch to a subcontractor with a proven reliability track record even though they are more expensive.
The interface with the new control device is not defined and as a result its driver may take more time to implement then scheduled.	Negotiate with the customer to move the implementation of this control device into a future software release. Replace the selected control device with an older device that has a well-defined interface.

Do we know enough? If we don't know enough, we can plan to "buy" additional information through mechanisms such as prototyping, modeling, simulation, or conducting additional research.

Risk	Obtain Additional Information
The subcontractor may not deliver the software at the required reliability level and as a result the reliability of the total system may not meet performance specifications.	Perform a capability assessment of the subcontractor. Ask for references from past customers and check on the reliability of previous products.
The interface with the new control device is not defined and as a result its driver may take more time to implement then scheduled.	Establish a communications link with device provider to obtain early design specifications for the device. Research interface specifications for other similar control devices by the same provider.

Can we transfer the risk? If it is not our risk or if it is economically feasible to pay someone else to assume all or part of the risk, we can plan to transfer the risk to another organization. For example we can contract with a disaster recovery firm to provide backup computer facilities that will allow continuation of the project in case a fire or other disaster destroys the project's work environment.

Risk	Transfer the Risk
The subcontractor may not deliver the software at the required reliability level and as a result the reliability of the total system may not meet performance specifications.	Transfer the risk to the subcontractor by building penalties into the contract for delivered software that does not have the required reliability.
The interface with the new control device is not defined and as a result its driver may take more time to implement then scheduled.	Transfer the risk to the customer by building additional charges to the customer and/or late delivery alternatives into the contract if the customer does not supply the specification by its due date.

Copyright © 2001 The Westfall Team. All Rights Reserved

Should action be taken now? If we decide to attack the risk directly, we typically start with creating a list of possible risk reduction actions that can be taken for the risk. Two major types of risk reduction actions should be considered:

1. Actions that reduce the likelihood that the risk will occur
2. Actions that reduce the impact of the risk should it occur

These may include actions such as establishing a liaison with the customer to insure adequate communications, conducting a performance simulation, or buying additional equipment for the test bed to duplicate the operational environment.

Risk	Control the Risk / Containment Plan
The subcontractor may not deliver the software at the required reliability level and as a result the reliability of the total system may not meet performance specifications.	Assign a project engineer to participate in the requirements & design inspection and to conduct alpha testing at the subcontractor's site. Require defect data reports from the subcontractor on a weekly basis during integration and system test.
The interface with the new control device is not defined and as a result its driver may take more time to implement then scheduled.	Assign a senior software engineer who has experience with similar control devices to the design task. Move the design task later in the schedule and increase its effort estimate.

From the list of possible risk reduction actions, we must select those that we are actually going to implement. When considering which risk reduction activities to select, a cost/benefit analysis should be performed. Boehm defines the Risk Reduction Leverage equation to help quantitatively establish the cost/benefit of implementing a risk reduction action. [Boehm-89]. Risk Reduction Leverage measures the return on investment of the available risk reduction techniques. Risk Reduction Leverage (RRL) is defined as the difference between the Risk Exposure before and after the reduction activity divided by the cost of that activity.

$$RRL = (RE_{before} - RE_{after}) / \text{Risk Reduction Cost}$$

If the Risk Reduction Leverage is less than one, it means that the cost of the risk reduction activity outweighs the probable gain from implementing the action.

Contingency plans: If immediate risk reduction actions are not taken or if those actions reduce but do not eliminate the risk, it may be appropriate to develop risk contingency plans. Contingency plans are plans that are implemented only if the risk actually turns into a problem. Examples of contingency plans include disaster recovery plans, contacting a consulting firm to establish a fallback position if key personnel are not available, or selecting an alternative design approach if the new technology is not delivered as promised.

Each risk that has a contingency plan should also have a trigger. A trigger is a time or event in the future that is the earliest indication that the risk will turn into a problem. For example, if there is a risk that outsourced software will not be delivered on schedule, the trigger could be whether the critical design review was held on schedule. A trigger can also be a relative variance or threshold metric. For example, if the risk is the availability of key personnel for the coding phase, the trigger could be a relative variance of more than 10% between actual and planned staffing levels.

By default, all of the risks that didn't have a high enough priority to warrant action planning are risks that we have assumed. We should also consider assigning triggers to these risks as indicators that detect status or priority changes. For example, because the probability of both the test bed and simulator hardware not being delivered in time for System Test is so small, we did not select this risk for action. However, we may want to set a trigger at the Integration Test Readiness Review to reexamine that risk.

Risks that are assigned triggers can be set at a "monitor only" status until the trigger occurs. At that time, the risk analysis step should be repeated to determine if risk reduction action is needed or if the contingency plan should be implemented.

Copyright © 2001 The Westfall Team. All Rights Reserved

There are trade-offs in utilizing triggers in risk management. We want to set the trigger as early as possible in order to ensure that there is plenty of time to implement risk reduction actions. We also want to set the trigger as late as possible because the longer we wait, the more information we have to make a correct decision and not implement unnecessary actions.

Risk	Assume the Risk / Contingency Plans
The subcontractor may not deliver the software at the required reliability level and as a result the reliability of the total system may not meet performance specifications.	Risk Assumption: This is the best subcontractor for the job and we will trust them to deliver reliable software. Early Trigger (reassessment of risk indicated): Completion of Critical Design Review (CDR) later than June 1st. Contingency Plan Trigger: More that 2 critical and 25 major defects detected during second pass of system test. Contingency Plan: Assign an engineer to liaison with the subcontractor on defect resolution and implement our regression test plan for maintenance releases from the subcontractor.
The interface with the new control device is not defined and as a result its driver may take more time to implement then scheduled.	Risk Assumption: This new technology will greatly improve the usability of the system. We will assume that the interface definition will be available by the time we are ready to design the driver. Early Trigger (reassessment of risk indicated): Interface definition not received by start of Preliminary Design Review (PDR). Contingency Plan Trigger: Interface definition not received by start of CDR. Contingency Plan: Hold CDR with a "To Be Done" and hold a second CDR for just the subsystem that uses the device.

Adjusting the project plan: Each selected action in the risk handling plans must include a description of the action and a list of tasks with assigned responsibilities and due dates. These actions must be integrated into the project plan with effort and cost estimations. Project schedules must be adjusted to include these new actions. For example, new tasks such as creating prototypes, doing research or conducting alpha testing at the customer's site must be included in the project plan.

TAKING ACTION

During the action step, we implement the risk reduction plan. Individuals execute their assigned tasks. Project effort estimations are adjusted to take into consideration additional effort needed to perform risk reduction activities or to account for projected additional effort if the risk turns into a problem. Some tasks may be moved forward in the schedule to ensure adequate time to deal with problems if they occur. Other tasks may be moved back in the schedule to allow time for additional information to be obtained. Budgets must also be adjusted to consider risk reduction activities.

Copyright © 2001 The Westfall Team. All Rights Reserved

TRACKING

Results and impacts of the risk reduction implementation must be tracked. The tracking step involves gathering data, compiling that data into information, and then reporting and analyzing that information. This includes measuring known risks and monitoring triggers, as well as measuring the impacts of risk reduction activities. The results of the tracking can be:

- Identification of new risks that need to be added to the risk list.
- Validation of known risk resolutions so risks can be removed from the risk list because they are no longer a threat to project success.
- Information that dictates additional planning requirements
- Implementation of contingency plan

Many of the software metrics typically used to manage software projects can also be used to track risks. For example, Gantt charts, earned value measures, and budget and resource metrics can help identify and track risks involving variances between plans and actual performance. Requirements churn, defect identification rates, and defect backlogs can be used to track rework risks, risks to the quality of the delivered product, and even schedule risks.

CONCLUSIONS

With ever-increasing complexity and increasing demand for bigger, better, and faster, the software industry is a high-risk business. When teams don't manage risk, they leave projects vulnerable to factors that can cause major rework, major cost or schedule over-runs, or complete project failure. Adopting a Software Risk Management Program is a step every software manager can take to more effectively manage software development initiatives. Risk management is an ongoing process that is implemented as part of the initial project planning activities and utilized throughout all of the phases of the software development lifecycle. Risk management requires a fear-free environment where risks can be identified and discussed openly. Based on a positive, proactive approach, risk management can greatly reduce or even eliminate the need for crisis management within our software projects.

REFERENCES

[Boehm-89] Barry W. Boehm, *Tutorial: Software Risk Management*, Les Alamitos, CA, IEEE Computer Society, 1989.

[Down-94] Alex Down, Michael Coleman, Peter Absolon, *Risk Management for Software Projects*, London, McGraw-Hill Book Company, 1994.

[Gilb-88] Tom Gilb, *Principles of Software Engineering Management*, Wokingham, England: Addison-Wesley, 1988.

[Ould-90] Martyn Ould, *Strategies For Software Engineering: The Management of Risk and Quality*, Chichester, England, John Wiley & Sons, 1990.

[SEI-93] Marvin J. Carr, Suresh L. Konda, Ira Monarch, F. Carol Ulrich, Clay F. Walker, Taxonomy-Based Risk Identification, CMU/SEI-93-TR-006, Pittsburgh, PA, Software Engineering Institute, 1993.

Copyright © 2001 The Westfall Team. All Rights Reserved