Know Your Enemy: Software Risk Management[1]

Karl E. Wiegers

Process Impact
716-377-5110
www.processimpact.com

Software engineers are eternal optimists. When planning software projects, we often assume that everything will go exactly as planned. Or, we take the other extreme position: the creative nature of software development means we can never predict what's going to happen, so what's the point of making detailed plans? Both of these perspectives can lead to software surprises, when unexpected things happen that throw the project off track. In my experience, software surprises are never good news.

Risk management is becoming recognized as a best practice in the software industry for reducing the surprise factor. While we can never predict the future with certainty, we can apply structured risk management practices to peek over the horizon at the traps that might be looming, and take actions to minimize the likelihood or impact of these potential problems. Risk management means dealing with a concern before it becomes a crisis. This improves the chance of successful project completion and reduces the consequences of those risks that cannot be avoided.

What Is Risk?

A simple definition of a "risk" is a problem that could cause some loss or threaten the success of our project, but which hasn't happened yet. (And we'd like to keep it that way.) These potential problems might have an adverse impact on the cost, schedule, or technical success of the project, the quality of our products, or team morale. Risk management is the process of identifying, addressing, and eliminating these potential problems before they can damage our project.

Whether we tackle them head-on or keep our heads in the sand, risks have a potentially huge impact on many aspects of our project. The tacit assumption that nothing unexpected will derail our project is simply not realistic. Our estimates should incorporate our best judgment about the potentially scary things that could happen on each project, and managers need to respect the assessments we make. Risk management is about discarding the rose-colored glasses and confronting the very real potential of undesirable events conspiring to throw our project off track.

Why Manage Risks Formally?

A formal risk management process provides a number of benefits to both the project team and the development organization as a whole. First, it gives us a structured mechanism to provide visibility into threats to project success. By considering the potential impact of each risk item, we

[1] This paper was originally published in *Software Development*, October 1998. It is reprinted (with modifications) with permission from *Software Development* magazine.

can focus on controlling the most severe risks first. We can combine risk assessment with project estimation to quantify possible schedule slippage if certain risks materialize into problems, thereby coming up with sensible contingency buffers. Sharing what does and does not work to control risks across multiple projects helps projects avoid repeating the mistakes of the past. Without a formal approach, we cannot ensure that our risk management actions will be initiated in a timely fashion, completed as planned, and effective.

Controlling risks has a cost, which we must balance against the potential cost we could incur if the risk is not addressed and does indeed bite us. For example, if we are concerned about the ability of a subcontractor to deliver an essential component on time, we could engage multiple subcontractors to increase the chance that at least one will come through on schedule. That's an expensive remedy for a problem that may not even exist. Is it worth it? It depends on the down side we incur if indeed the subcontractor dependency causes the project to miss its planned ship date. Only you can decide for each individual situation.

Typical Software Risks

The list of evil things that can befall a software project is depressingly long. The enlightened project manager will acquire extensive lists of these risk categories to help the team uncover as many concerns as possible early in the planning process. Possible risks to consider can come from group brainstorming activities, or from a risk factor chart accumulated from previous projects. In one group I've worked in, individual team members came up with insightful descriptions of their risk factors, which I edited together and we then reviewed as a team.

The Software Engineering Institute (SEI) has assembled a taxonomy of hierarchically-organized risks in 13 major categories, with about 200 thought-provoking questions to help you spot the risks facing your project. These are listed in SEI Technical Report CMU/SEI-93-TR-006, "Taxonomy-Based Risk Identification," by Marvin Carr, et al. Steve McConnell's Jolt Award-winning *Rapid Development* (Microsoft Press, 1996) also contains excellent resource material on risk management.

Following are several typical risk categories and risk items that may threaten your project. Have any of these things have happened to you? If so, add them to your master risk factor checklist to remind future project managers to ask themselves if it could happen to them, too. There are no magic solutions to any of these risk factors, so we need to rely on past experience and a strong knowledge of software engineering and management practices to control those risks we are most concerned about.

Dependencies

Many risks arise because of dependencies our project has on outside agencies or factors. We cannot usually control these external dependencies, so mitigation strategies may involve contingency plans to acquire a necessary component from a second source, or working with the source of the dependency to maintain good visibility into status and detect any looming problems. Here are some typical dependency-related risk factors:

- customer-furnished items or information
- internal and external subcontractor relationships
- inter-component or inter-group dependencies
- availability of trained, experienced people

- reuse from one project to the next

Requirements Issues

Many projects face uncertainty and turmoil around the product’s requirements. While some of this uncertainty is tolerable in the early stages, the threat to success increases if such issues are not resolved as the project progresses. If we don’t control requirements-related risk factors, we might either build the wrong product, or build the right product badly. Either situation results in unpleasant surprises and unhappy customers. Watch out for these risk factors:

- lack of a clear product vision
- lack of agreement on product requirements
- inadequate customer involvement in the requirements process
- unprioritized requirements
- new market with uncertain needs
- rapidly changing requirements
- ineffective requirements change management process
- inadequate impact analysis of requirements changes

Management Issues

Although management shortcomings inhibit the success of many projects, don’t be surprised if your risk management plan doesn’t list very many of these. After all, the project manager is usually the person who is writing the risk management plan, and most people don’t wish to air their own weaknesses (assuming they even recognize them) in public. Nonetheless, issues like those listed here can make it harder for projects to succeed. If we don’t confront such touchy issues, we shouldn’t be surprised if they bite us at some point. Defined project tracking processes, and clear project roles and responsibilities, can address some of these risk factors.

- inadequate planning and task identification
- inadequate visibility into actual project status
- unclear project ownership and decision making
- unrealistic commitments made, sometimes for the wrong reasons
- managers or customers with unrealistic expectations
- staff personality conflicts

Lack of Knowledge

The rapid rate of change of software technologies, and the increasing shortage of skilled staff, mean our project teams may not have the skills we need to be successful. The key is to recognize the risk areas early enough so that we can take appropriate preventive actions, such as obtaining training, hiring consultants, and bringing the right people together on the project team. These factors might apply to your team:

- lack of training
- inadequate understanding of methods, tools, and techniques
- insufficient application domain experience
- new technologies or development methods
- ineffective, poorly documented, or ignored processes
- technical approaches that may not work

Risk Management Approaches

Risk management is the application of appropriate tools and procedures to contain risk within acceptable limits. It consists of several sub-activities.

- ***Risk assessment*** is the process of examining a project and identifying areas of potential risk. ***Risk identification*** can be facilitated with the help of a checklist of common risk areas for software projects, or by examining the contents of an organizational database of previously identified risks and mitigation strategies (both successful and unsuccessful)..
  Risk analysis involves examining how project outcomes might change with modification of risk input variables.
  Risk prioritization helps the project focus on its most severe risks by assessing the risk exposure. Exposure is the product of the probability of incurring a loss due to the risk and the potential magnitude of that loss. I usually estimate the probability from 0.1 (highly unlikely) to 1.0 (certain to happen), and the loss on a relative scale of 1 (no problemo) to 10 (deep tapioca). Multiplying these factors together provide an estimation of the risk exposure due to each item, which can run from 0.1 (don’t give it another thought) through 10 (stand back, here it comes!). It may be easier to simply estimate both probability and impact as High, Medium, or Low. Those items having at least one dimension rated as High are the ones to worry about first.

- ***Risk avoidance*** is one way to deal with a risk: don’t do the risky thing! You may avoid risks by not undertaking certain projects, or by relying on proven rather than cutting edge technologies when possible.

- ***Risk control*** is the process of managing risks to achieve the desired outcomes. ***Risk management planning*** produces a plan for dealing with each significant risk, including mitigation approaches, owners, and timelines. ***Risk resolution*** is execution of the plans for dealing with each risk. Finally, ***risk monitoring*** involves tracking your progress toward resolving each risk item.

Simply identifying the risks facing your project is not enough. We need to write them down in a way that lets us communicate the nature and status of risk factors throughout the affected stakeholder community over the duration of the project. Figure 1 shows a form I have found to be convenient for documenting risks; this information can also be stored in a spreadsheet. This risk list could be included as a section in your software project plan, or it could remain as a standalone document.

Figure 1. Risk Documentation Form.

<table>
<tr><td colspan="6"><b>ID:</b> (sequence number is fine)</td></tr>
<tr><td colspan="6"><b>Description:</b> (List each major risk facing the project. Describe each risk in the form “condition – consequence”.)</td></tr>
<tr><td colspan="2"><b>Probability:</b> (What’s the likelihood of this risk becoming a problem?)</td><td colspan="2"><b>Loss:</b> (What’s the damage if the risk does become a problem?)</td><td colspan="2"><b>Exposure:</b> (Multiply Probability times Loss to estimate the risk exposure.)</td></tr>
<tr><td colspan="6"><b>First Indicator:</b> (Describe the earliest indicator or trigger condition that might indicate that the risk is turning into a problem.)</td></tr>
<tr><td colspan="6"><b>Mitigation Approaches:</b> (State one or more approaches to control, avoid, minimize, or otherwise mitigate the risk.)</td></tr>
<tr><td colspan="3"><b>Owner:</b> (Assign each risk mitigation action to an individual for resolution.)</td><td colspan="3"><b>Date Due:</b> (State a date by which the mitigation approach is to be implemented.)</td></tr>
</table>

When documenting risk statements, use a *condition-consequence* format. That is, state the risk situation (condition) that you are concerned about, followed by at least one potential adverse outcome (consequence) if that risk should turn into a problem. Often, people suggesting risks may state only the condition (“the customers don’t agree on the product requirements”) or the consequence (“we can only satisfy one of our major customers”). Pull those together into the condition-consequence structure: “The customers don’t agree on the product requirements, so we’ll only be able to satisfy one of our major customers.”

The P, L, and E cells in the form provide locations to capture the probability of a risk materializing into a problem (P), the loss we could incur as a result of that problem (L), and the overall risk exposure (P times L). Keep the high-exposure items at the top of your priority list. You can’t address every risk item, so use this prioritization mechanism to learn where to focus your risk control energy. Set goals for determining when each risk item has been satisfactorily controlled. For some items, your mitigation strategies may focus on reducing the probability, while the approach for other items may emphasize reducing the loss.

The cell labeled Risk Mitigation Approaches allows you to identify the actions you intend to take to keep the risk item under control. With any luck, some of your mitigation approaches can be used to address multiple risk factors. For example, one group I’ve worked with identified several risks related to failures of components of their web delivery infrastructure (servers, firewall, e-mail interface, and so on). A mitigation strategy that addressed several of those risks was to implement an automated monitoring system that could check the status of the servers and communication functions periodically and alert us to any failures.

As with other project management activities, you need to get into a rhythm of periodic monitoring. You may wish to appoint a “risk czar” for the project, who is responsible for staying on top of the things that could go wrong, just as the project manager is staying on top of the activities leading to project completion. One company I know of assigned an individual to such a role, and dubbed him "Eeyore," after the Winnie-the-Pooh character who always bemoaned how bad things could become.

Keep the top ten or so risks highly visible, and track the effectiveness of your mitigation approaches regularly. As the initial list of top priority items gradually gets beaten into submission, new items may float up into the top ten. Don't kid yourself into concluding that a risk is controlled simply because the selected mitigation action has been completed. Controlling a risk may mean that you have to change your mitigation strategy if you conclude it is ineffective. You can drop the risk off your threat-detection radar when you determine that your mitigation approaches have indeed reduced the loss exposure from that item to an acceptable level.

Risk Management Can Be Your Friend

The skillful project manager will use risk management as a technique for raising the awareness of conditions that could cause her project to go down the tubes. Consider a project that begins with an unclear product vision and a lack of customer involvement. The astute project manager will spot this situation as posing potential risks, and will document them in the risk management plan. Early in the project's life, the impact of this situation may not be too severe. However, if time continues to pass and the lack of product vision and customer involvement are not improved, the risk exposure will steadily rise.

By reviewing the risk management plan periodically, the project manager can adjust the probability and/or impact of these risks. Those that don't get controlled can be brought to the attention of senior managers or other stakeholders who are in a position to either stimulate corrective actions, or make a conscious business decision to proceed in spite of the risks. We're keeping our eyes open and making informed decisions, even if we can't control every adverse condition facing our project.

Learning from the Past

While we can't predict exactly which of the many threats to our projects might come to pass, most of us can do a better job of learning from previous experiences to avoid the same pain and suffering on future projects. As you begin to implement risk management approaches, keep records of your actions and results for future reference. Try these suggestions:

- Record the results of even informal risk assessments, to capture the thinking of the project participants.

- Document the mitigation strategies attempted for each risk you chose to confront, noting which approaches worked well and which did not pay off.

- Conduct post-project reviews to identify the unanticipated problems that arose. Should you have been able to see them coming through a better risk management approach, or would you likely have been blindsided in any case? Do you think these same problems might occur on other projects? If so, add them to your growing checklist of potential risk factors that the next project can think about.

Anything you can do to improve your ability to avoid or minimize problems on future projects will improve your company's business success. Risk management can also reduce the chaos, frustration, and constant fire-fighting that reduces the quality of work life in so many software organizations. The risks are out there. Find them before they find you.