Chapter 5
Planning for Software Project Risks

Learning Objectives of this Chapter:

✓ Identifying project risks

✓ Several software risks to avoid

✓ Completing qualitative and quantitative analyses

✓ Choosing a software development model

✓ Preparing the risk response plan

Risks

- R*isk is an uncertain event that could have a positive or negative outcome.*

- Risk is everywhere: whether you’re driving, sky diving, crossing the street, trading stocks, or swimming with sharks.

- When you do something risky, you must calculate whether the potential reward is worth the potential risk.

Project Managers Risk

➢ One of the toughest jobs for any project manager is managing project risks.

➢ In addition to risks that affect your ability to complete your assignments, project risks include the following:

- Inadequate time for completing the project
- Inadequate budget for completing the project
- Unrealistic scope expectations
- A project team that needs additional time to ramp up development
language
- Stakeholders that do not or cannot provide clear project requirements

Identifying Pure and Business Risks

➢ When most people think of risks, they have an immediate negative connotation about risk.

➢ The risk, however, isn’t what makes folks frown; it’s the impact of the risk. Risk itself is not really a bad thing.

➢ Positive risks are called *risk opportunities.*

➢ You can’t possibly consider every single risk in a project. All projects deal with risks that are usually so far off the risk radar that they aren’t even a concern:

❖ The company might go out of business
❖ An asteroid could crash into the office building
❖ Big Foot could appear and take all the back-up tapes
❖ The senior developer might move to Hawaii

See? These risks are unusual, highly improbable, and are way, way, out there in left field. But all of these are risks — you just have to accept them. As a project manager, you must consider the risk result, and you also have to categorize risks.

Identifying Pure and Business Risks

In life and project management there are two types of risks to be concerned with:

Pure risks: These risks have no upside, only a downside. Pure risks include things like loss of life or limb, fire, flood, and other bad stuff that nobody likes.

Business risks: These risks are the calculated risks you are concerned with in project management. A perfect example of a business risk is using a worker with less experience in order to save money on the project’s budget.

Assessing business risks

Business risks are a big concern for any project manager with an eye towards reality. Business risks are the more common risks you encounter in your project management activities:

- ✓ Employees quit
- ✓ Mistakes are made in the requirements gathering process
- ✓ The software is full of bugs, errors, and failures
- ✓ The scope of the project grows, but the budget (or the timeline) doesn’t
- ✓ The expectations of the project time, cost, and scope are not realistic to begin with
- ✓ The project is larger than the capacity of the project team
- ✓ The project manager, sponsor, or other stakeholders are not as knowledgeable as you would hope

Accepting everyday technology risks with your software project

Consider these risks in every software project:

- ✓ Speed of technology surpasses demand for your creation.
- ✓ Delays in your schedule shorten the window for the demand of your software.
- ✓ Your programmers' ability to learn new programming languages and adapt to new development environments may threaten the project schedule.
- ✓ Your stakeholders may have a tough time explaining what they want the project deliverable to be.
- ✓ Because programmers are in demand, a programmer could leave your project team, putting your project at risk from loss of talent, time away from progress, and time devoted to getting a new developer up to speed.
- ✓If your project has never been attempted before, you risk suffering from the first-time, first-use penalty. This penalty basically means that because it's never been done before there is a higher risk of facing problems you couldn't possibly anticipate.

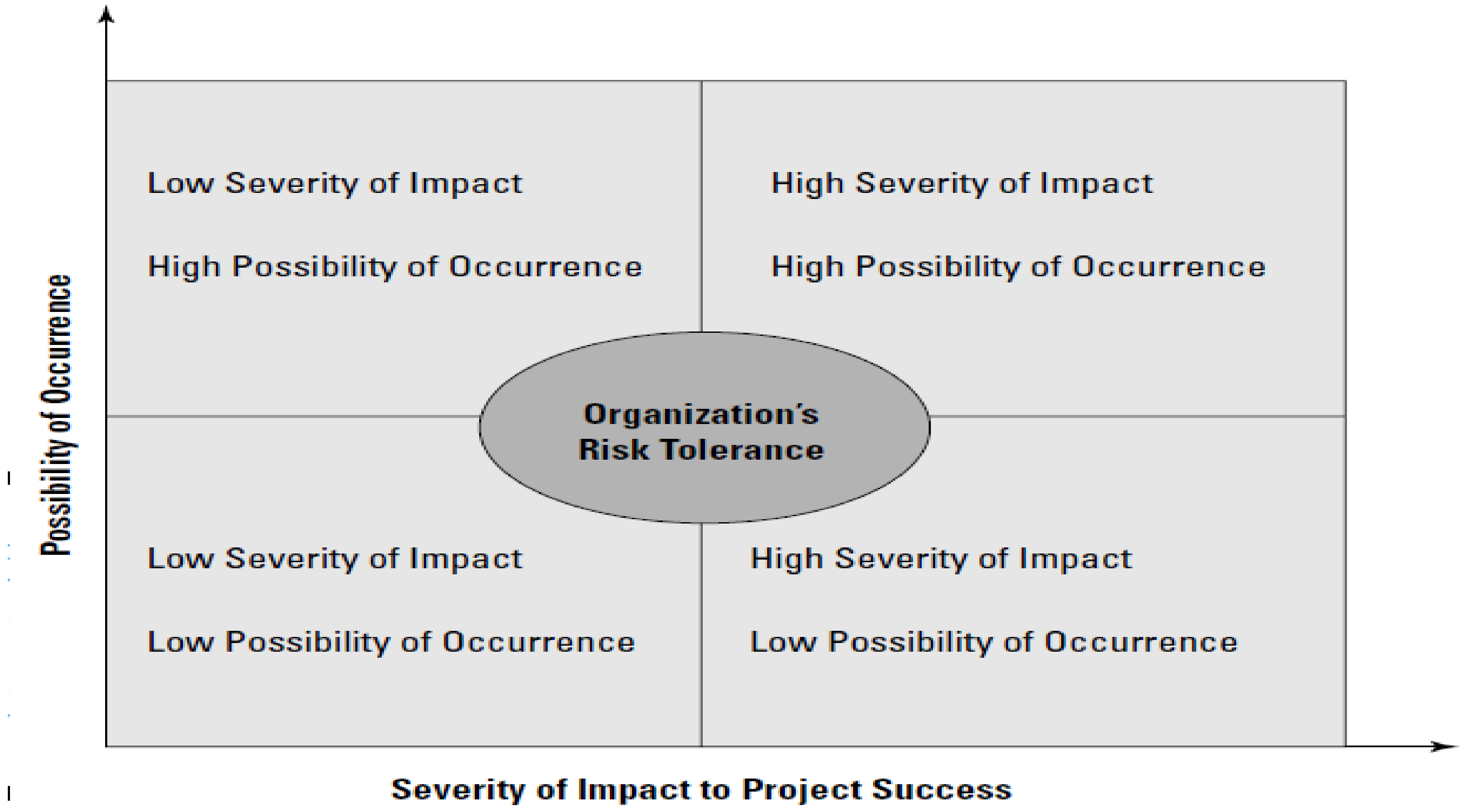
Determining Stakeholder Risk Tolerance

- A person’s willingness to accept risks is called his or her *utility function.*

- Your willingness to invest in a riskier venture, and the amount you’re comfortable investing, describe your utility function. The same theory applies to your stakeholders. You, your project team, and the key stakeholders will be happy to accept some risks and will refuse to accept others.

- Figure 5-2 shows an S-curve. As you can see, the higher the project priority is the lower the utility function is. In other words, the higher the project priority, the more likely you are to reduce risks. The amount of acceptable risks will diminish.

Fig 5.1: Project Priorities Determines a Stakeholders Utility Function


Low Severity of Impact
High Possibility of Occurrence
High Severity of Impact
High Possibility of Occurrence
Organization's Risk Tolerance
Low Severity of Impact
Low Possibility of Occurrence
High Severity of Impact
Low Possibility of Occurrence
Possibility of Occurrence
Severity of Impact to Project Success

Mitigating Risks Early On

Fig 5.2: Projects are more likely to fail at the beginning than at the end.

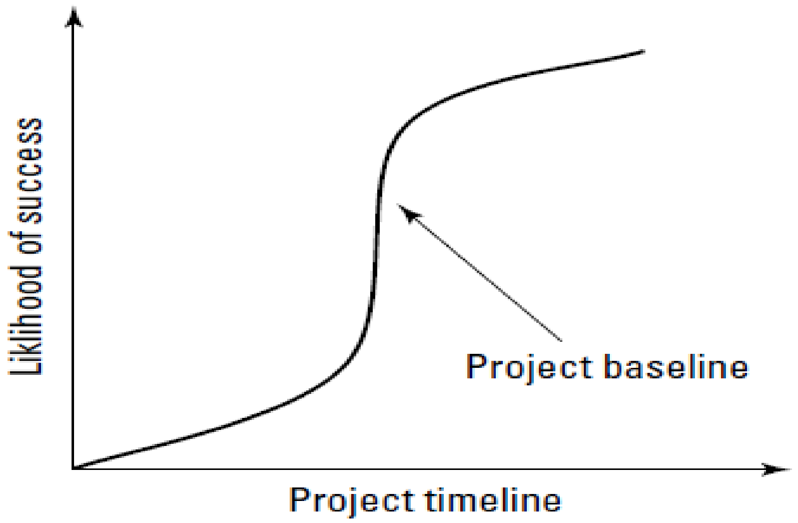


Figure 5-2 shows that as the project moves closer to the end of development, it's more likely to succeed.

Managing Risks in Your Organization

• Every organization has its own approach to risk management.

• Tighter organizations have a step-by-step strategy to handling risks. These organizations have procedures, templates, and processes to identify, capture, and asses the risks that threaten the project's success.

Identifying risks

• Regardless of your company's official risk management strategy, you can rely on ***qualitative risk analysis*** *to get things moving. Qualitative analysis is the* process of creating a risk ranking based on all the identified risks within the project.

• The best way to conduct qualitative risk analysis is for you to invite the project team and all the other key stakeholders to get together for a risk identification party.

• The method you use to gather information and identify risks is not as important as the fact that you are obtaining as much information as possible. For these risk identification exercises, put the emphasis on quantity. More is better. If a risk is not identified, it is accepted by default.

Managing Risks in Your Organization

- **Brainstorming**

- **Following the Delphi method**: This allows stakeholders to anonymously offer their input into identifying risks. They can send their suggestions via e-mail to one person who will then consolidate the information into one document — without naming the source of the information.

Ranking risks

After you and the key stakeholders identify all the risks you can think of, you need to rank them. We suggest ranking project risks by using a *risk impact matrix.*

You can use two different approaches to risk ranking:

Ordinal: This assessment simply ranks risks as high, medium, or low. Most folks use ordinal for the first round of risk analysis.
Cardinal: When you use this ranking system, you assign scores with hard numbers, like .67 or .99.
With each risk, you and your project team, and sometimes the key stakeholders, need to follow these steps:

1. Evaluate the probability of the risk actually happening and assign that risk a score.
2. Score the impact of each risk.
3. Multiply the probability and the impact to get a risk score.

Risk Rating Matrix

Table 5-1 gives you a quick example of a risk rating matrix. In this example, we used an ordinal scoring method because it's a bit more tangible.

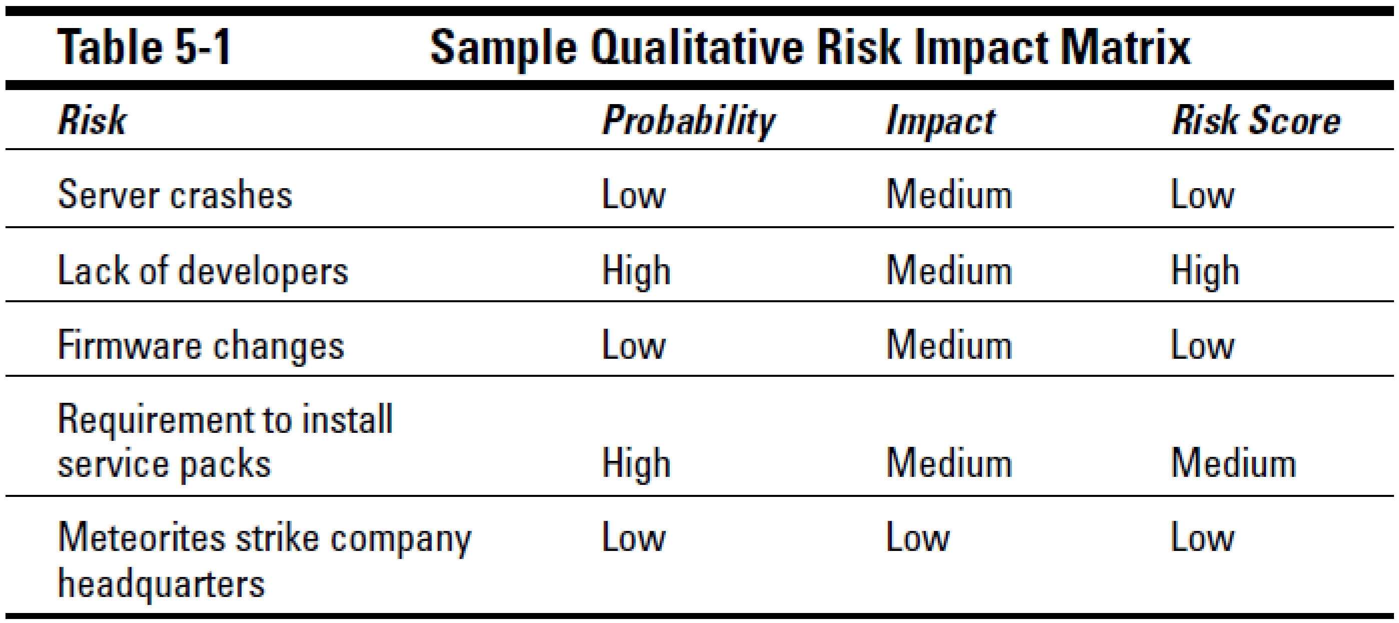


Table 5-1 Sample Qualitative Risk Impact Matrix

Risk	*Probability*	*Impact*	*Risk Score*
Server crashes	Low	Medium	Low
Lack of developers	High	Medium	High
Firmware changes	Low	Medium	Low
Requirement to install service packs	High	Medium	Medium
Meteorites strike company headquarters	Low	Low	Low

Risk in Software Projects

Eight risks every software project has:

- ✓Time
- ✓Costs
- ✓Scopes
- ✓Feasibility
- ✓Quality
- ✓Stakeholders expectations
- ✓Human Resources
- ✓Technical Accuracy

Relying on Quantitative Analysis

- ***Quantitative analysis** is the process of measuring your risk exposure.*

- *Quantification* requires more than going with your gut feeling; you need to conduct interviews, set up prototypes, do expert analysis, and set up simulations.

- As you can guess, quantitative analysis takes time — and usually some investment.

- Someone has to interview the stakeholders. Someone has to create the prototype, the simulations, and complete the analysis.

- Some stakeholders, customers, and project managers are likely to argue against completing a quantitative analysis because of the time and cost involved to do it correctly. It's up to you to convince them that it's worth the extra effort.

- The investment in completing the interviews, prototypes, and simulations is generally much smaller than responding to the crushing effects on a project's time and cost baselines if a risk is realized.

Difference between qualitative and quantitative analyses

- **Qualitative analysis** means that you’re describing the qualities of the risks; quantitative analysis gives a quantity, usually a number, and often with a dollar sign next to it, of the impacts of those risks.

- **Qualitative analysis** is better for smaller projects (under $100,000 and shorter than three months). Larger projects, with more money at stake and longer durations, require more quantitative analysis.

Creating a Contingency Reserve

- **Quantitative analysis** also uses a risk impact matrix, like qualitative analysis, though a *cardinal scale is mostly used here*. This risk impact matrix also quantifies the dollars or time the project stands to lose, or gain, because of the risk.
- What the project needs is a contingency reserve that will alleviate the expenses of the risks should they come into fruition. You know how much to set aside for this contingency reserve by using the quantitative risk impact matrix.

- **Table 5-2 gives you an idea of what a matrix might reveal.**

Table 5.2: Sample Quantitative Risk Impact Matrix

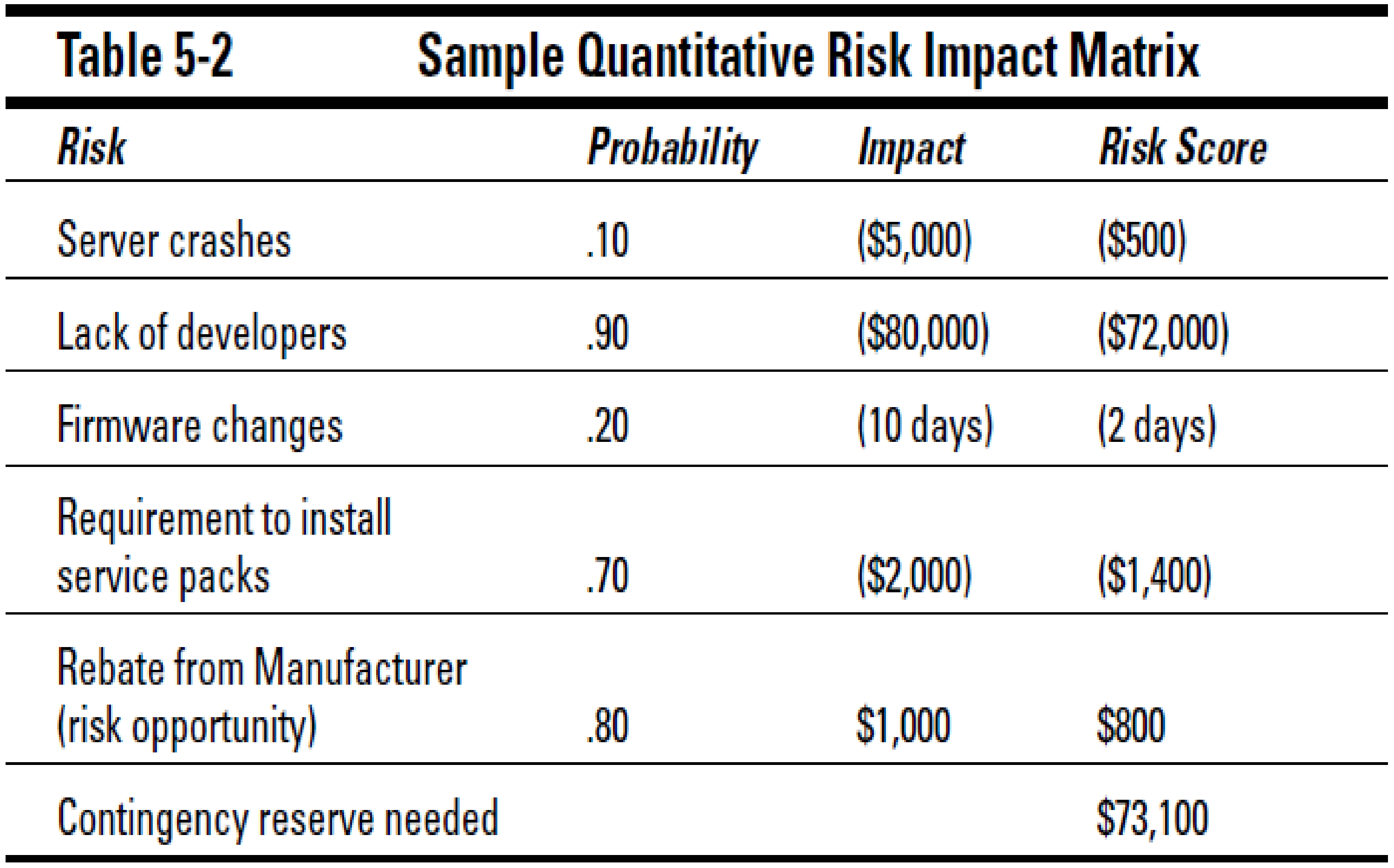


Table 5-2 Sample Quantitative Risk Impact Matrix

Risk	*Probability*	*Impact*	*Risk Score*
Server crashes	.10	($5,000)	($500)
Lack of developers	.90	($80,000)	($72,000)
Firmware changes	.20	(10 days)	(2 days)
Requirement to install service packs	.70	($2,000)	($1,400)
Rebate from Manufacturer (risk opportunity)	.80	$1,000	$800
Contingency reserve needed			$73,100

Table 5.2: Sample Quantitative Risk Impact Matrix

- The impact of each risk is quantified by a negative dollar amount or by an assessment of time lost, though some organizations turn time into dollars. This negative dollar amount represents the cost the project will incur if the risk event occurs. Notice that the rebate from the manufacturer risk will actually save the project $1,000 if it comes to fruition.

- The sum of the risk event values, both positive and negative, is the amount of contingency funds that should be reserved for this project. Don’t get too excited that the amount of the contingency reserve is less than the sum of all the risks. The probability for all of the risks occurring is not 100 percent, so you won’t get 100 percent of the dollar amount associated with each risk. You’re banking that some of the risk events will occur and some of the events won’t.

Using Software Models for Risk Management

- Within software project management, **there are models** that help the project manager alleviate the biggest risk of them all — failure. Models are adapted from grandfather programs, serious thought and planning, or they evolve based on past experience. Whatever approach you or your organization takes, the model needs to be documented, and the rules need to be defined and then followed.
- The following sections describe the most popular software models on a continuum from ***risk-enriched models*** *(high-risk* approaches) to ***risk-averse models*** *(relatively risk-free approaches).*

Using the waterfall model

• Within software project management, there are models that help the project manager alleviate the biggest risk of them all — failure. Models are adapted from grandfather programs, serious thought and planning, or they evolve based on past experience. Whatever approach you or your organization takes, the model needs to be documented, and the rules need to be defined and then followed.

• The following sections describe the most popular software models on a continuum from *risk-enriched models (high-risk* approaches) to *risk-averse models (relatively risk-free approaches).*

•The ***waterfall model*** *uses a series* of phases to move the project along. Each phase creates a deliverable, usually a document that captures what the phase has accomplished. Figure 5-3 shows the progression of the model.

Fig 5.3: The waterfall model follows a series of phases to reach completion

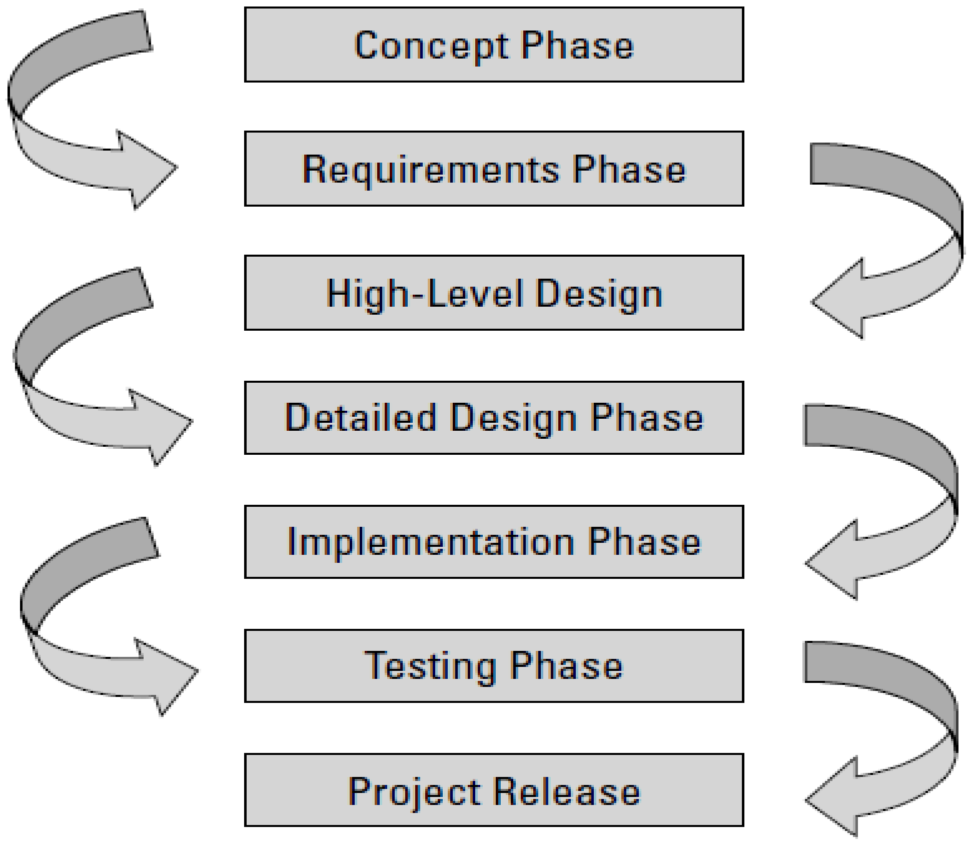


Fig 5.3: The waterfall model follows a series of phases to reach completion

With the completion of each phase comes a new stage, with the product of the phase transitioning (just like a waterfall) into the next phase. Here’s the progression:

1. **Come up with a concept**
2. **Determine the requirements**
3. **Create a high-level design**
4. **Narrow down the design to create a detailed design**
5. **Code implementation phase**
6. **Testing phase**

At this stage, you test the entire application to ensure that all the units are coded and working as expected — before your customers see it. This is the quality control phase. If you find problems, then you’ve got more work to do. The goal of this phase, of course, is to keep mistakes out of the customers’ hands. When all is well, the deliverable is released.

Using the Spiral Model

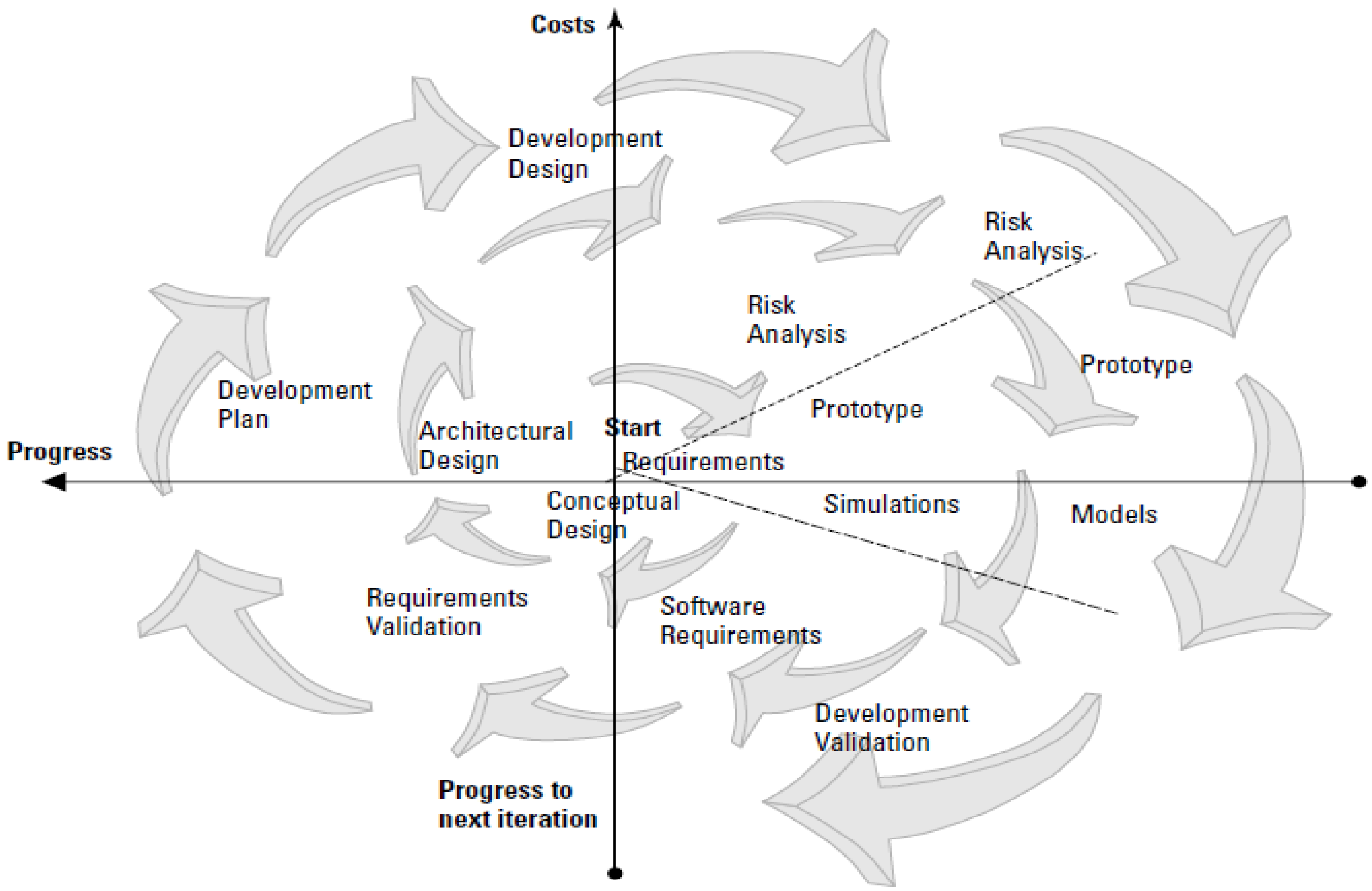
- **The Spiral model** is the safest, or most risk adverse, model available. An organization that uses the spiral model examines the project as a whole and then breaks down the project into subprojects, each of which are categorized by risks. The subprojects are then organized from risk heavy to risk lean.

- With the spiral model, you tackle the areas of the project where most of the toughest risks are first. This approach, which is frankly our favorite, hits the project risks head-on and then moves on to the next risk-laden subproject.

- Take a look at Figure 5-4; see how the project starts at the center and spirals out like a cinnamon roll? The completion of each subproject enables the project to move on to the next subproject until the project spirals all the way out to the release.

Fig: 5.4 The spiral model uses iterations to move the project to completion.

•



Spiral Model

Here’s the general approach:

- Set a goal.
- Conduct alternative investigations
- Conduct risk management
- Create the deliverables
- Plan the next iteration
- Determine what to do next

Based on the previous step, you determine what needs to happen next. After you make that determination, you move all the way back to step one and move through these steps with the next subproject.

Using the V model

- This model is called the V model because, well, because the progression of the project forms a V.

- Look at Figure 5-5; see the V? This is a risk-averse development model because the completion of each phase prompts a corresponding test phase. Before the project moves forward, the test must be passed.

- This model is technically an expansion of the waterfall model, with added verification and testing. Like the waterfall model, each phase creates documentation of what’s been completed in the project.

Fig 5.4: The V model uses a series of tests for each completed phase to move the project forward

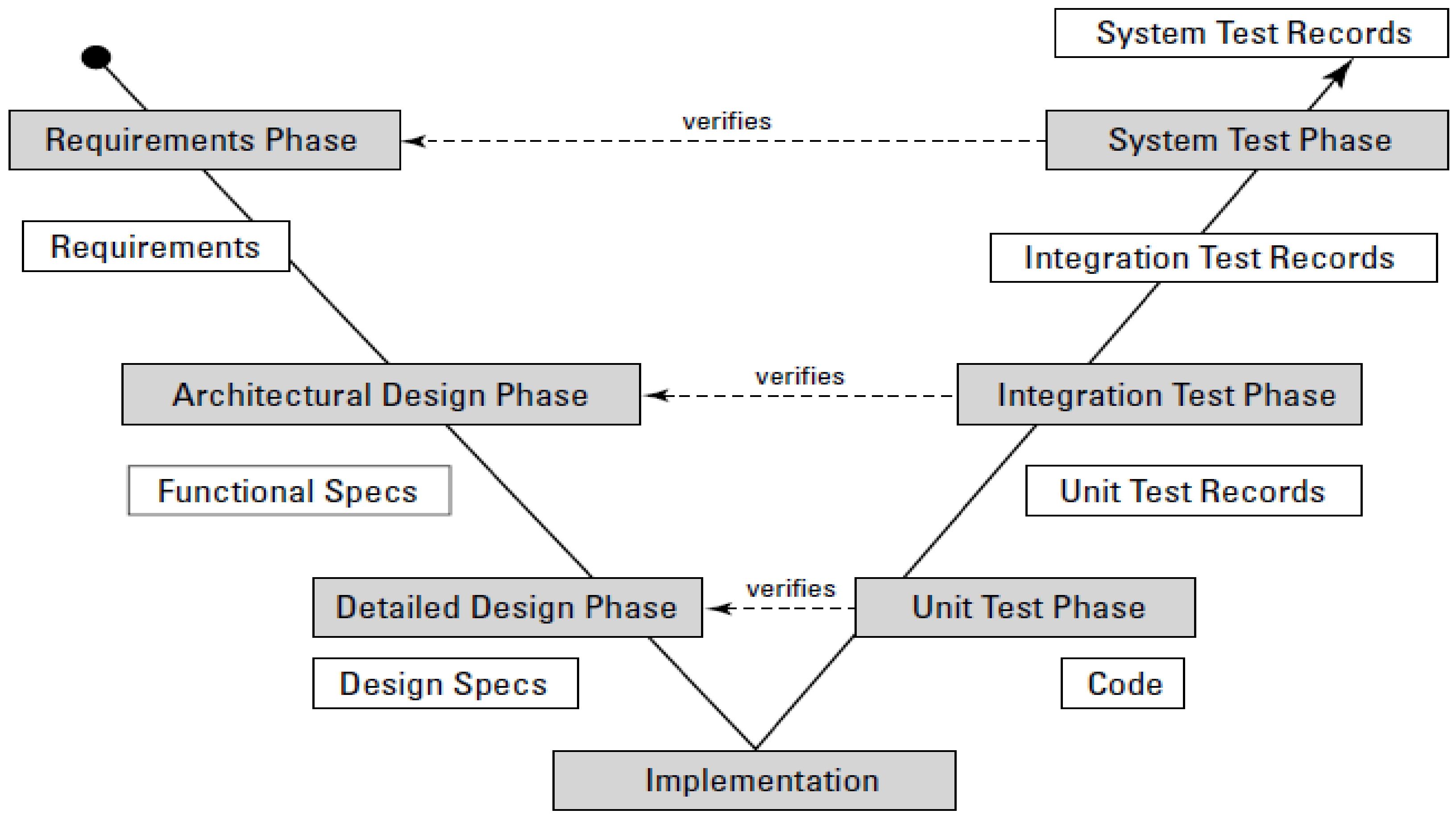


Fig 5.4: The V model uses a series of tests for each completed phase to move the project forward

Here’s a breakdown of the V model progression:

1. **Set the project requirements**
2. **Design the architecture**
3. **Elaborate on the architecture with a detailed design**
4. **Implement the code**
5. **Test individual units**
6. **Test how everything works together**
7. **Test the whole system**

In the system test phase, compile the software and test the system as a whole. Successful testing of this phase allows the system to be released.

Using the Scrum development model

Scrum, in software development, means working in quick iterations, building team empowerment, and being adaptable. Figure 5-6 illustrates what the scrum model looks like.

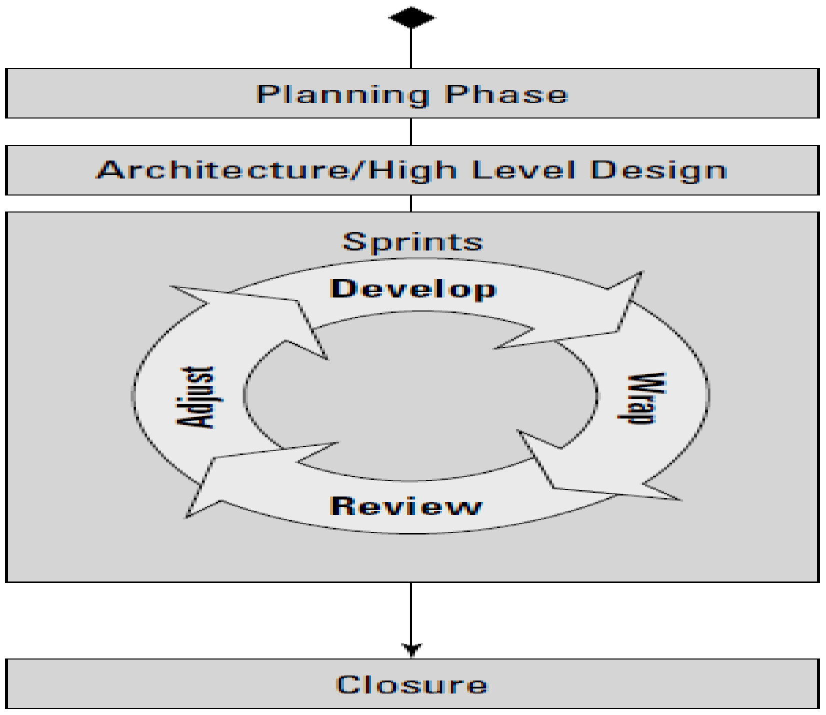


Fig: 5.6: Scrum relies on sprints for bursts of software development

Using the Scrum development model

- As in rugby, the scrum development model has lots of rules that the project team and the project manager must live by.
- The first key rule in scrum is to *never interrupt the programmers while they're working.*
- *The second crucial* rule is that everyone must follow the same process for work prioritization.
- In addition, scrum depends on solid communication, collocated teams, and quick, accurate team meetings. Here's how it works:

Set up a plan

Design the architecture

Start sprinting

Each sprint includes the following features:

Development

Wrap

Review

Adjustment

Close out the project

During the closure phase, the project results are tested and deemed accurate. The software is then prepped for release.

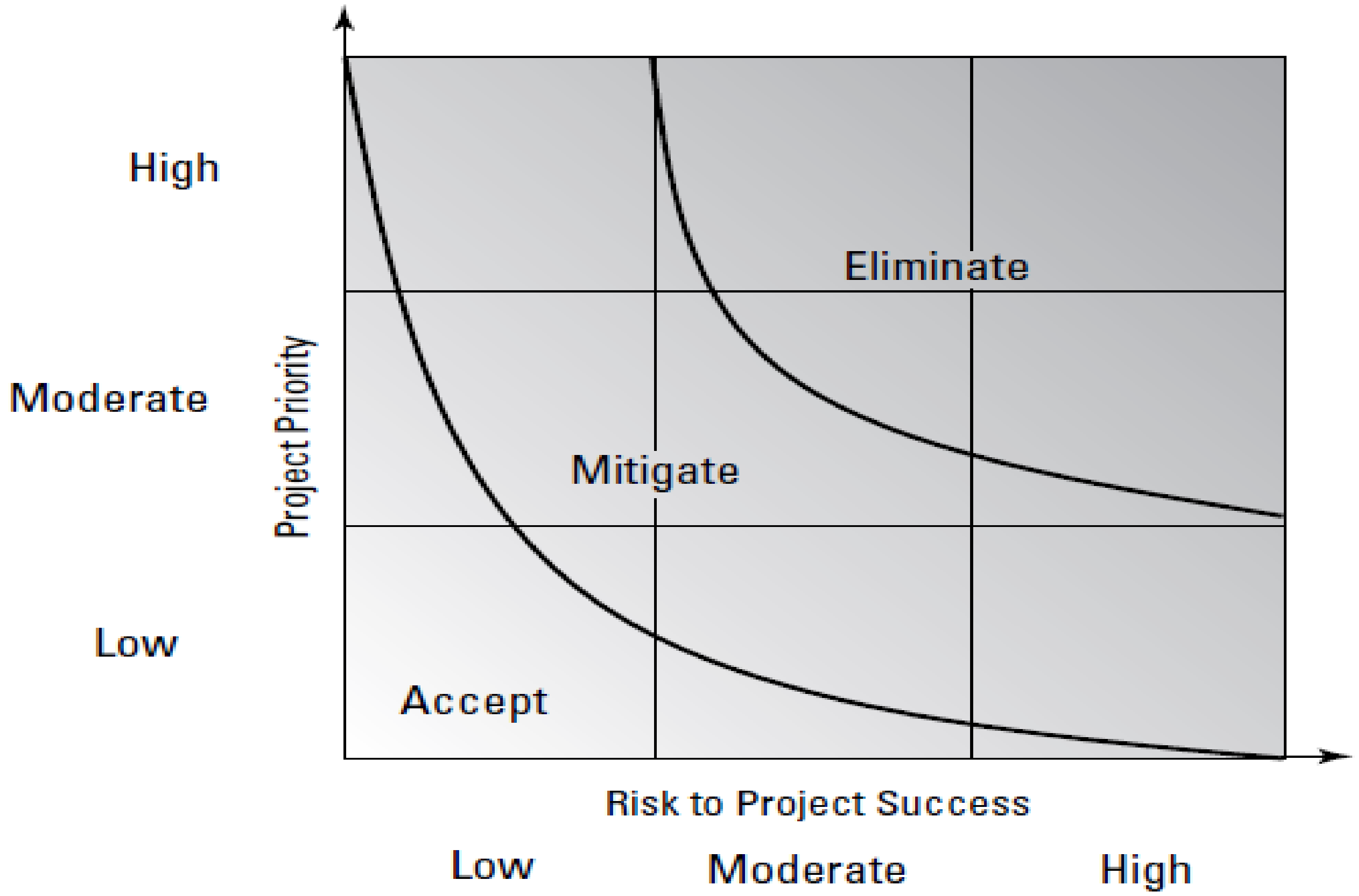
Preparing a Risk Response Plan

- Every project, regardless of the software development model you use, has project risks. As the expert project manager, you (along with your project team) surely have identified the project risks, used quantitative and qualitative risk analyses, and scored and ranked your risks. But is this enough? Of course not.

- Typically you, the project team, and the project stakeholders do an initial qualitative risk analysis. That's the listing and ranking of your project risks.

- Figure 5-7 shows **the scale of the risks** you can usually live with, the risks with impacts you want to reduce, and the risks you want to work to eliminate. The *risk response plan helps you come to these decisions.*
- **The risk response plan** is a document that details the identified risks within a project, their impact, and their associated costs, and then identifies how the project team will respond to the risks.
- In addition, the risk response plan nods to the process of risk management. Risk identification, the first step of risk management, is an iterative process that happens throughout the project —not just at the beginning.

Figure 5-7: Your risk response plan helps you see that not all risks need to be eliminated



Avoiding risks

You’ve done risk avoidance if you’ve done any of the following:

- Changed a project plan to avoid risk interruption
- Used an established approach to software development rather than a newfangled model
- Hired experts to consult the project team during the development process
- Spent additional time with the project stakeholders to clarify all project objectives and requirements

Transferring risks

***Transference** means that the risk doesn’t go away. It’s just someone else’s* responsibility now. You’ve used transference if you’ve ever done any of the following:

✓Purchased insurance, such as errors and omissions insurance
✓ Hired experts to complete a portion of the project work
✓ Demanded warranties from vendors
✓ Brought in consultants to test units and builds of your software

Mitigating risks

• ***Risk mitigation*** *is about reducing the impact and/or the probability of risk.*

You've used mitigation if you've ever done the following:

• Added extra testing, verification, or customer approval activities to ensure that the software conforms to requirements
• Reduced the number of processes, activities, or interactions within a smaller project to streamline and focus project management activities on accomplishing specific project tasks
• Developed and tested prototypes, used user acceptability testing processes, or launched pilot groups within your organization before releasing the software

Examining Risk Responses and Impacts

The project team and the project manager should determine when the risk response should be implemented. Two terms here to recognize are:

- **Risk threshold:** The line of demarcation that signals that a risk is about to come into play and that some response should happen. The risk threshold can be a date for completion, a percentage of the work that is not complete, a failed test, or any other event that signals a pending risk.

- **Risk trigger:** A trigger is an event within the project that triggers a preplanned response to the identified risk.

Handling the ripple effect of risk response

There are two key risks that come from risk responses that should be examined with every risk response:

Residual risks: Residual risks are usually tiny risks that linger after a risk response. These are generally accepted and the project moves forward

Secondary risks: Secondary risks are more serious. They occur when a risk response creates significant new project risks.

Thanks