This chapter discusses how organizations evaluate and select projects to undertake from the many available projects. Once a project has been selected, the project manager plans the project. Project management involves selecting a project methodology, creating the project work plan, identifying project staffing requirements, and preparing to manage and control the project. These steps produce important project management deliverables, including the work plan, staffing plan, standards list, project charter, and risk assessment.

OBJECTIVES

- Explain how projects are selected in some organizations.
- Describe various approaches to the SDLC that can be used to structure a development project.
- Explain how to select a project methodology based on project characteristics.
- Become familiar with project estimation.
- Be able to create a project work plan.
- Describe project staffing issues and concerns.
- Describe and apply techniques to coordinate and manage the project.
- Explain how to manage risk on the project.

CHAPTER OUTLINE

Introduction
Project Selection
   Applying the Concept at Tune Source
Creating the Project Plan
   Project Methodology Options
   Selecting the Appropriate Methodology
   Estimating the Project Time Frame
   Developing the Work Plan
   Staffing the Project
   Coordinating Project Activities
Managing and Controlling the Project
   Refining Estimates
   Managing Scope
   Timeboxing
   Managing Risk
Applying the Concepts at Tune Source
Summary
Appendix 2A—The Function Point Approach
Appendix 2B—Project Management Tools: The Gantt Chart and PERT Chart
INTRODUCTION

Most IT departments face a demand for IT projects that far exceeds the department’s ability to supply them. In the past 10 years, business application growth has exploded, and chief information officers (CIOs) are challenged to select projects that will provide the highest possible return on IT investments while managing project risk. In a recent analysis, AMR Research Inc. found that 2%–15% of projects taken on by IT departments are not strategic to the business.¹ In today’s globally competitive business environment, the corporate IT department needs to carefully prioritize, select, and manage its portfolio of development projects.

Historically, IT departments have tended to select projects by ad hoc methods: first-in, first-out; political clout; or the squeaky wheel getting the grease. In recent years, IT departments have collected project information and mapped the projects’ contributions to business goals, using a project portfolio perspective.² Project portfolio management, a process of selecting, prioritizing, and monitoring project results, has become a critical success factor for IT departments facing too many potential projects with too few resources.³ Software for project portfolio management, such as Hewlett Packard’s Project and Portfolio Management, Primavera Systems’ ProSight, and open-source Project.net, has become a valuable tool for IT organizations.

Once selected, a systems development project undergoes a thorough process of project management, the process of planning and controlling the project within a specified time frame, at minimum cost, with the desired outcomes.⁴ A project manager has the primary responsibility for managing the hundreds of tasks and roles that need to be carefully coordinated. Project management has evolved into an actual profession with many training options and professional certification (e.g., Project Management Professional, or PMP) available through the Project Management Institute (www.pmi.org). Dozens of software products are available to support project management activities.

Although training and software are available to help project managers, unreasonable demands set by project sponsors and business managers can make project management very difficult. Too often, the approach of the holiday season, the chance at winning a proposal with a low bid, or a funding opportunity pressures project managers to promise systems long before they are realistically able to deliver them. These overly optimistic timetables are thought to be one of the biggest problems that projects face; instead of pushing a project forward faster, they result in delays.

Thus, a critical success factor for project management is to start with a realistic assessment of the work that needs to be accomplished and then manage the project according to the plan. This can be accomplished by carefully following the basic steps of project management as outlined in this chapter. First, the project manager chooses a system development methodology that fits the characteristics of the project. Based on the size of the system, estimates of a time frame are made. Then, a list of tasks to be performed is created that forms the basis of the project work plan. Staffing needs are determined, and the project manager sets in place mechanisms to coordinate the project team throughout the project. Finally, the project manager monitors the project and refines estimates as work proceeds.

² Ibid.
⁴ A good book on project management is by Robert K. Wysocki, Effective Project Management: Traditional, Adaptive, Extreme, 5th Ed., New York: John Wiley & Sons, 2009. Also, the Project Management Institute (www.pmi.org) and the Information Systems Special Interest Group of the Project Management Institute (www.pmi-issig.org) have valuable resources on project management in information systems.
Many IT organizations tackle a number of important initiatives simultaneously. For example, new software applications may be under development; new business models may be under consideration; organizational structures may be revised; new technical infrastructures may be evaluated. Collectively, these endeavors are managed as a program by the IT steering committee. The steering committee must provide oversight and governance to the entire set of projects that are undertaken by the IT organization. The individual projects that are accepted by the steering committee are temporary endeavors undertaken to create a unique product or service.

Investments in information systems projects today are evaluated in the context of an entire portfolio of projects. Decision makers look beyond project cost and consider a project’s anticipated risks and returns in relation to other projects. Companies prioritize their business strategies and then assemble and assess project portfolios on the basis of how they meet those strategic needs.

The focus on a project’s contribution to an entire portfolio of projects reinforces the need for the feasibility study as described in Chapter 1. The approval committee has the responsibility to evaluate not only the project’s costs and expected benefits, but also the technical and organizational risks associated with the project. The feasibility analysis is submitted back to the approval committee, along with an updated system request. Using this information, the approval committee can examine the business need (found in the system request) and the project risks (described in the feasibility analysis).

Portfolio management takes into consideration the different kinds of projects that exist in an organization—large and small, high risk and low risk, strategic and tactical.
A good project portfolio will have the most appropriate mix of projects for the organization’s needs. The committee acts as a portfolio manager, with the goal of maximizing benefits versus costs and balancing other important factors of the portfolio. For example, an organization may want to keep high-risk projects to a level less than 20% of its total project portfolio. The approval committee must be selective about where to allocate resources, because the organization has limited funds. This involves trade-offs in which the organization must give up something in return for something else in order to keep its portfolio well balanced. If there are three potentially high-payoff projects, yet all have very high risk, then maybe only one of the projects will be selected. Also, there are times when a system at the project level makes good business sense, but it does not at the organization level. Thus, a project may show a very strong economic feasibility and support important business needs for a part of the company; however, it is not selected. This could happen for many reasons—because there is no money in the budget for another system, the organization is about to go through some kind of change (e.g., a merger, an implementation of a company-wide system like an ERP), projects that meet the same business requirements already are underway, or the system does not align well with current or future corporate strategy.

Applying the Concepts at Tune Source

The approval committee met and reviewed the Digital Music Download project along with two other projects—one that called for a new supply-chain portal and another that involved the enhancement of Tune Source’s data warehouse. Unfortunately, the budget would allow for only one project to be approved, so the committee carefully examined

<table>
<thead>
<tr>
<th>Size</th>
<th>What is the size? How many people are needed to work on the project?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cost</td>
<td>How much will the project cost the organization?</td>
</tr>
<tr>
<td>Purpose</td>
<td>What is the purpose of the project? Is it meant to improve the technical infrastructure? support a current business strategy? improve operations? demonstrate a new innovation?</td>
</tr>
<tr>
<td>Length</td>
<td>How long will the project take before completion? How much time will go by before value is delivered to the business?</td>
</tr>
<tr>
<td>Risk</td>
<td>How likely is it that the project will succeed or fail?</td>
</tr>
<tr>
<td>Scope</td>
<td>How much of the organization is affected by the system? a department? a division? the entire corporation?</td>
</tr>
<tr>
<td>Economic Value</td>
<td>How much money does the organization expect to receive in return for the amount the project costs?</td>
</tr>
</tbody>
</table>

(See Figure 2-1 for different ways of classifying projects.)
In April 1999, one of Capital Blue Cross’ health-care insurance plans had been in the field for three years, but hadn’t performed as well as expected. The ratio of premiums to claims payments wasn’t meeting historic norms. In order to revamp the product features or pricing to boost performance, the company needed to understand why it was underperforming. The stakeholders came to the discussion already knowing they needed better extraction and analysis of usage data in order to understand product shortcomings and recommend improvements.

After listening to input from the user teams, the stakeholders proposed three options. One was to persevere with the current manual method of pulling data from flat files via ad hoc reports and retyping it into spreadsheets. The second option was to write a program to dynamically mine the needed data from Capital’s customer information control system (CICS). While the system was processing claims, for instance, the program would pull out up-to-the-minute data at a given point in time for users to analyze.

The third alternative was to develop a decision-support system to allow users to make relational queries from a data mart containing a replication of the relevant claims and customer data.

Each of these alternatives was evaluated on cost, benefits, risks, and intangibles.

QUESTION:
1. What are three costs, benefits, risks, and intangibles associated with each project?
2. Based on your answer to question 1, which project would you choose?


A CIO needs to have a global view when identifying and selecting projects for her organization. I would get lost in the trees if I were to manage on a project-by-project basis. Given this, I categorize my projects according to my three roles as a CIO, and the mix of my project portfolio changes depending on the current business environment.

My primary role is to keep the business running. That means every day when each person comes to work, they can perform his or her job efficiently. I measure this using various service level, cost, and productivity measures. Projects that keep the business running could have a high priority if the business were in the middle of a merger, or a low priority if things were running smoothly, and it were “business as usual.”

My second role is to push innovation that creates value for the business. I manage this by looking at our lines of business and asking which lines of business create the most value for the company. These are the areas for which I should be providing the most value. For example, if we had a highly innovative marketing strategy, I would push for innovation there. If operations were running smoothly, I would push less for innovation in that area.

My third role is strategic, to look beyond today and find new opportunities for both IT and the business of providing energy. This may include investigating process systems, such as automated meter reading or looking into the possibilities of wireless technologies.

Lyn McDermid
At Marriott, we don’t have IT projects—we have business initiatives and strategies that are enabled by IT. As a result the only time a traditional “IT project” occurs is when we have an infrastructure upgrade that will lower costs or leverage better functioning technology. In this case, IT has to make a business case for the upgrade and prove its value to the company.

The way IT is involved in business projects in the organization is twofold. First, senior IT positions are filled by people with good business understanding. Second, these people are placed on key business committees and forums where the real business happens, such as finding ways to satisfy guests. Because IT has a seat at the table, we are able to spot opportunities to support business strategy. We look for ways in which IT can enable or better support business initiatives as they arise.

Therefore, business projects are proposed, and IT is one component of them. These projects are then evaluated the same as any other business proposal, such as a new resort—by examining the return on investment and other financial measures.

At the organizational level, I think of projects as must-do’s, should-do’s, and nice-to-do’s. The “must-do’s” are required to achieve core business strategy, such as guest preference. The “should-do’s” help grow the business and enhance the functionality of the enterprise. These can be somewhat untested, but good drivers of growth. The “nice-to-do’s” are more experimental and look further out into the future.

The organization’s project portfolio should have a mix of all three kinds of projects, with a much greater proportion devoted to the “must-do’s.”

Carl Wilson

Hygeia Travel Health is a Toronto-based health insurance company whose clients are the insurers of foreign tourists to the United States and Canada. Its project selection process is relatively straightforward. The project evaluation committee, consisting of six senior executives, splits into two groups. One group includes the CIO, along with the heads of operations and research and development, and it analyzes the costs of every project. The other group consists of the two chief marketing officers and the head of business development, and they analyze the expected benefits. The groups are permanent, and to stay objective, they don’t discuss a project until both sides have evaluated it. The results are then shared, both on a spreadsheet and in conversation. Projects are then approved, passed over, or tabled for future consideration.

Last year, the marketing department proposed purchasing a claims database filled with detailed information on the costs of treating different conditions at different facilities. Hygeia was to use this information to estimate how much money insurance providers were likely to owe on a given claim if a patient was treated at a certain hospital as opposed to any other. For example, a 45-year-old man suffering a heart attack may accrue $5000 in treatment costs at hospital A, but only $4000 at hospital B. This information would allow Hygeia to recommend the less expensive hospital to its customer. That would save the customer money and help differentiate Hygeia from its competitors.

The benefits team used the same three-meeting process to discuss all the possible benefits of implementing the claims database. Members of the team talked to customers and made a projection by using Hygeia’s past experience and expectations about future business trends. The verdict: The benefits team projected a revenue increase of $210,000. Client retention would rise by 2%, and overall, profits would increase by 0.25%.

The costs team, meanwhile, came up with large estimates: $250,000 annually to purchase the database and an additional $71,000 worth of internal time to make the information usable. Put it all together and it was a financial loss of $111,000 in the first year.

The project still could have been good for marketing—maybe even good enough to make the loss acceptable. But some of Hygeia’s clients were also in the claims information business and, therefore, potential competitors. This, combined with the financial loss, was enough to make the company reject the project.

Creating the Project Plan

Once the project is launched by being selected by the approval committee, it is time to carefully plan the project. The project manager will follow a set of project management guidelines, sometimes referred to as the project management life cycle, as he or she organizes, guides, and directs the project from inception to completion. Generally speaking, the project management phases consist of initiation, planning, execution, control, and closure.

In large organizations or on large projects, the role of project manager is commonly filled by a professional specialist in project management. In smaller organizations or on smaller projects, the systems analyst may fill this role. The project manager must make a myriad of decisions regarding the project, including determining the best project methodology, developing a work plan for the project, determining a staffing plan, and establishing mechanisms to coordinate and control the project.

Project Methodology Options

As we discussed in Chapter 1, the Systems Development Life Cycle (SDLC) provides the foundation for the processes used to develop an information system. A methodology is a formalized approach to implementing the SDLC (i.e., it is a list of steps and deliverables). There are many different systems development methodologies, and they vary in terms of the progression that is followed through the phases of the SDLC. Some methodologies are formal standards used by government agencies, while others have been developed by consulting firms to sell to clients. Many organizations have their own internal methodologies that have been refined over the years, and they explain exactly how each phase of the SDLC is to be performed in that company. Here we will review several of the predominant methodologies that have evolved over time.

Waterfall Development  With waterfall development, analysts and users proceed sequentially from one phase to the next. (See Figure 2-2.) The key deliverables for each phase are typically voluminous (often, hundreds of pages) and are presented
to the approval committee and project sponsor for approval as the project moves from phase to phase. Once the work produced in one phase is approved, the phase ends and the next phase begins. As the project progresses from phase to phase, it moves forward in the same manner as a waterfall. While it is possible to go backward through the phases (e.g., from design back to analysis), it is quite difficult. (Imagine yourself as a salmon trying to swim upstream in a waterfall.)

Waterfall development methodologies have the advantages of identifying requirements long before programming begins and limiting changes to the requirements as the project proceeds. The key disadvantages are that the design must be completely specified before programming begins, a long time elapses between the completion of the system proposal in the analysis phase and the delivery of system, and testing is treated almost as an afterthought in the implementation phase. In addition, the deliverables are often a poor communication mechanism, so important requirements may be overlooked in the volumes of documentation. If the project team misses an important requirement, expensive post-implementation programming may be needed. Users may forget the original purpose of the system, since so much time has elapsed between the original idea and actual implementation. Also, in today’s dynamic business environment, a system that met the existing environmental conditions during the analysis phase may need considerable rework to match the environment when it is implemented. This rework requires going back to the initial phase and making needed changes through each of the subsequent phases in turn.

**Figure 2-3**
Parallel Development
There are two important variants of waterfall development. The *parallel development* methodologies evolved to address the lengthy time frame of waterfall development. As shown in Figure 2-3, instead of doing the design and implementation in sequence, a general design for the whole system is performed. Then the project is divided into a series of subprojects that can be designed and implemented in parallel. Once all subprojects are complete, there is a final integration of the separate pieces, and the system is delivered.

Parallel development reduces the time required to deliver a system, so changes in the business environment are less likely to produce the need for rework. The approach still suffers from problems caused by voluminous deliverables. It also adds a new problem: If the subprojects are not completely independent, design decisions in one subproject may affect another, and at the project end, integrating the subprojects may be quite challenging.

The *V-model* is another variation of waterfall development that pays more explicit attention to testing. As shown in Figure 2-4, the development process proceeds down the left-hand slope of the V, defining requirements and designing system components. At the base of the V, the code is written. On the upward-sloping right side of the model, testing of components, integration testing, and, finally, acceptance testing are performed. A key concept of this model is that as requirements are specified and components designed, testing for those elements is also defined. In this manner, each level of testing is clearly linked to a part of the analysis or design phase, helping to ensure high quality and relevant testing and maximize test effectiveness.
The V-model is simple and straightforward and improves the overall quality of systems through its emphasis on early development of test plans. Testing focus and expertise is involved in the project earlier rather than later; plus, the testers gain knowledge of the project early. It still suffers from the rigidity of the waterfall development process, however, and is not always appropriate for the dynamic nature of the business environment.

Rapid Application Development (RAD)\textsuperscript{5} Rapid application development is a collection of methodologies that emerged in response to the weaknesses of waterfall development and its variations. RAD incorporates special techniques and computer tools to speed up the analysis, design, and implementation phases in order to get some portion of the system developed quickly and into the hands of the users for evaluation and feedback. CASE (computer-aided software engineering) tools, JAD (joint application development) sessions, fourth-generation/visual programming languages (e.g., Visual Basic.NET), and code generators may all play a role in RAD. While RAD can improve the speed and quality of systems development, it may also introduce a problem in managing user expectations. As systems are developed more quickly and users gain a better understanding of information technology, user expectations may dramatically increase and system requirements may expand during the project (sometimes known as scope creep or feature creep).

RAD may be conducted in a variety of ways. Iterative development breaks the overall project into a series of versions that are developed sequentially. The most important and fundamental requirements are bundled into the first version of the system. This version is developed quickly by a mini-waterfall process, and once implemented, the users can provide valuable feedback to be incorporated into the next version of the system. (See Figure 2-5.) Iterative development gets a preliminary version of the system to the users quickly so that business value is provided. Since users are working with the system, important additional requirements may be identified and incorporated into subsequent versions. The chief disadvantage of iterative development is that users begin to work with a system that is intentionally incomplete. Users must accept that only the most critical requirements of the system will be available in the early versions and must be patient with the repeated introduction of new system versions.

System prototyping performs the analysis, design, and implementation phases concurrently in order to quickly develop a simplified version of the proposed system and give it to the users for evaluation and feedback. (See Figure 2-6). The system prototype is a “quick and dirty” version of the system and provides minimal features. Following reaction and comments from the users, the developers reanalyze, redesign, and reimplement a second prototype that corrects deficiencies and adds more features. This cycle continues until the analysts, users, and sponsor agree that the prototype provides enough functionality to be installed and used in the organization. System prototyping very quickly provides a system for users to evaluate and reassures users that progress is being made. The approach is very useful when users have difficulty expressing requirements for the system. A disadvantage, however, is the lack of careful, methodical analysis prior to making design and implementation decisions. System prototypes may have some fundamental design limitations that are a direct result of an inadequate understanding of the system’s true requirements early in the project.

\textsuperscript{5} One of the best RAD books is that by Steve McConnell, \textit{Rapid Development}, Redmond, WA: Microsoft Press, 1996.
FIGURE 2-5
Iterative Development

FIGURE 2-6
System Prototyping
Throwaway prototyping\(^6\) includes the development of prototypes, but uses the prototypes primarily to explore design alternatives rather than as the actual new system (as in system prototyping). As shown in Figure 2-7, throwaway prototyping has a fairly thorough analysis phase that is used to gather requirements and to develop ideas for the system concept. Many of the features suggested by the users may not be well understood, however, and there may be challenging technical issues to be solved. Each of these issues is examined by analyzing, designing, and building a design prototype. A design prototype is not intended to be a working system. It contains only enough detail to enable users to understand the issues under consideration.

For example, suppose that users are not completely clear on how an order entry system should work. The analyst team might build a series of HTML pages to be viewed on a Web browser to help the users visualize such a system. In this case, a series of mock-up screens appear to be a system, but they really do nothing. Or, suppose that the project team needs to develop a sophisticated graphics program in Java. The team could write a portion of the program with artificial data to ensure that they could create a full-blown program successfully.

A system that is developed by this type of methodology probably requires several design prototypes during the analysis and design phases. Each of the prototypes is used to minimize the risk associated with the system by confirming that important issues are understood before the real system is built. Once the issues are resolved, the project moves into design and implementation. At this point, the design prototypes are thrown away, which is an important difference between this approach and system prototyping, in which the prototypes evolve into the final system.

Throwaway prototyping balances the benefits of well-thought-out analysis and design phases with the advantages of using prototypes to refine key issues before a system is built. It may take longer to deliver the final system compared with System.

system prototyping (because the prototypes do not become the final system), but
the approach usually produces more stable and reliable systems.

**Agile Development** Agile development is a group of programming-centric methodologies that focus on streamlining the SDLC. Much of the modeling and documentation overhead is eliminated; instead, face-to-face communication is preferred. A project emphasizes simple, iterative application development in which every iteration is a complete software project, including planning, requirements analysis, design, coding, testing, and documentation. (See Figure 2.8). Cycles are kept short (one to four weeks), and the development team focuses on adapting to the current business environment. There are several popular approaches to agile development, including extreme programming (XP), Scrum, and dynamic systems development method (DSDM). Here, we briefly describe extreme programming.

**Extreme programming** emphasizes customer satisfaction and teamwork. Communication, simplicity, feedback, and courage are core values. Developers communicate with customers and fellow programmers. Designs are kept simple and clean. Early and frequent testing provides feedback, and developers are able to courageously respond to changing requirements and technology. Project teams are kept small.

An XP project begins with user stories that describe what the system needs to do. Then, programmers code in small, simple modules and test to meet those needs. Users are required to be available to clear up questions and issues as they arise. Standards are very important to minimize confusion, so XP teams use a common set of names, descriptions, and coding practices. XP projects deliver results sooner than even the RAD approaches, and they rarely get bogged down in gathering requirements for the system.

For small projects with highly motivated, cohesive, stable, and experienced teams, XP should work just fine. However, if the project is not small or the teams aren’t

---

7 For more information, see www.AgileAlliance.org.
8 For more information, see www.extremeprogramming.org.
9 For more information, see www.controlchaos.com.
10 For more information, see www.dsdm.com
then the likelihood of success for the XP project is reduced. Consequently, the use of XP in combination with outside contractors produces a highly questionable outcome, since the outside contractors may never “jell” with insiders. XP requires a great deal of discipline to prevent projects from becoming unfocused and chaotic. Furthermore, it is recommended only for small groups of developers (not more than 10), and it is not advised for mission-critical applications. Since little analysis and design documentation is produced with XP, there is only code documentation; therefore, maintenance of large systems developed using XP may be impossible. Also, since mission-critical business information systems tend to exist for a long time, the utility of XP as a business information system development methodology is in doubt. Finally, the methodology requires considerable on-site user input, something that is frequently difficult to obtain.

### Agile versus Waterfall-Based Methodologies

Agile development approaches have existed for over a decade. Agile development practices were created in part because of dissatisfaction with the sequential, inflexible structure of waterfall-based approaches. Presently, agile development has made inroads into software development organizations, and studies show an even split between agile and waterfall users. Many organizations are experimenting with agile even while continuing to employ traditional waterfall approaches (see Concepts in Action 2F).

---

12 A “jelled team” is one that has low turnover, a strong sense of identity, a sense of eliteness, a feeling that they jointly own the product being developed, and enjoyment in working together. For more information regarding jelled teams, see T. DeMarco and T. Lister, *Peopleware: Productive Projects and Teams*, New York: Dorsett House, 1987.

13 Considering the tendency for offshore outsourcing, this is a major obstacle for XP to overcome. For more information on offshore outsourcing, see P. Thibodeau, “ITAA panel debates outsourcing pros, cons,” *Computerworld Morning Update*, September 25, 2003; and S. W. Ambler, “Chicken Little Was Right,” *Software Development*, October 2003.

14 Many of the observations described here on the utility of XP as a development approach are based on conversations with Brian Henderson-Sellers.

In fact, suggesting that an organization must be “all agile” or “all waterfall” is a false choice. Many software developers are actively seeking to integrate the best elements of both waterfall and agile into their software development practices. Hybrid agile-waterfall approaches are evolving. The process of developing information systems is never static. Most IS departments and project managers recognize that the choice of the “best” development methodology depends on project characteristics, as we discuss in the next section.

**Selecting the Appropriate Development Methodology**

As the previous section shows, there are many methodologies. The first challenge faced by project managers is to select which methodology to use. Choosing a methodology is not simple, because no one methodology is always best. (If it were, we’d simply use it everywhere!) Many organizations have standards and policies to guide the choice of methodology. You will find that organizations range from having one “approved” methodology to having several methodology options to having no formal policies at all.

Figure 2-9 summarizes some important methodology selection criteria. One important item not discussed in this figure is the degree of experience of the analyst team. Many of the RAD and agile development methodologies require the use of new tools and techniques that have a significant learning curve. Often, these tools and techniques increase the complexity of the project and require extra time for learning. Once they are adopted and the team becomes experienced, the tools and techniques can significantly increase the speed in which the methodology can deliver a final system.

**Clarity of User Requirements** When the user requirements for what the system should do are unclear, it is difficult to understand them by talking about them and explaining them with written reports. Users normally need to interact with technology to really understand what the new system can do and how to best apply it to their needs. System prototyping and throwaway prototyping are usually more appropriate when user requirements are unclear, because they provide prototypes for users to interact with early in the SDLC. Agile development may also be appropriate if on-site user input is available.
Familiarity with Technology  When the system will use new technology with which the analysts and programmers are not familiar (e.g., the first Web development project with Ajax), applying the new technology early in the methodology will improve the chance of success. If the system is designed without some familiarity with the base technology, risks increase because the tools may not be capable of doing what is needed. Throwaway prototyping is particularly appropriate for situations where there is a lack of familiarity with technology, because it explicitly encourages the developers to create design prototypes for areas with high risks. Iterative development is good as well, because opportunities are created to investigate the technology in some depth before the design is complete. While one might think that system prototyping would also be appropriate, it is much less so because the early prototypes that are built usually only scratch the surface of the new technology. Typically, it is only after several prototypes and several months that the developers discover weaknesses or problems in the new technology.

System Complexity  Complex systems require careful and detailed analysis and design. Throwaway prototyping is particularly well suited to such detailed analysis and design, but system prototyping is not. The waterfall methodologies can handle complex systems, but without the ability to get the system or prototypes into users’ hands early on, some key issues may be overlooked. Although iterative development methodologies enable users to interact with the system early in the process, we have observed that project teams who follow these methodologies tend to devote less attention to the analysis of the complete problem domain than they might if they were using other methodologies.

System Reliability  System reliability is usually an important factor in system development. After all, who wants an unreliable system? However, reliability is just one factor among several. For some applications, reliability is truly critical (e.g., medical equipment, missile control systems), while for other applications it is merely important (e.g., games, Internet video). The V-model is useful when reliability is important, due to its emphasis on testing. Throwaway prototyping is most appropriate when system reliability is a high priority, because detailed analysis and design phases are combined with the ability for the project team to test many different approaches through design prototypes before completing the design. System prototyping is generally not a good choice when reliability is critical, due to the lack of careful analysis and design phases that are essential to dependable systems.

2-3 Selecting a Methodology

Suppose that you are an analyst for the ABC Company, a large consulting firm with offices around the world. The company wants to build a new knowledge management system that can identify and track the expertise of individual consultants anywhere in the world on the basis of their education and the various consulting projects on which they have worked. Assume that this is a new idea that has never before been attempted in ABC or elsewhere. ABC has an international network, but the offices in each country may use somewhat different hardware and software. ABC management wants the system up and running within a year.

Question: What methodology would you recommend that ABC Company use? Why?
Short Time Schedules  Projects that have short time schedules are well suited for RAD methodologies because those methodologies are designed to increase the speed of development. Iterative development and system prototyping are excellent choices when time lines are short because they best enable the project team to adjust the functionality in the system on the basis of a specific delivery date. If the project schedule starts to slip, it can be readjusted by removal of the functionality from the version or prototype under development. Waterfall-based methodologies are the worst choice when time is at a premium, because they do not allow for easy schedule changes.

Schedule Visibility  One of the greatest challenges in systems development is knowing whether a project is on schedule. This is particularly true of the waterfall-based methodologies because design and implementation occur at the end of the project. The RAD methodologies move many of the critical design decisions to a position earlier in the project to help project managers recognize and address risk factors and keep expectations in check.

Estimating the Project Time Frame  As the previous section illustrated, some development methodologies have evolved in an attempt to accelerate the project through the SDLC as rapidly as possible while still producing a quality system. Regardless of whether time is a critical issue on a project or not, the project manager will have to develop a preliminary estimate of the amount of time the project will take. Estimation is the process of assigning projected values for time and effort.

Estimation can be performed manually or with the help of an estimation software package like Construx Estimate™, Costar™, or KnowledgePLAN®—there are over 50 available on the market. The estimates developed at the start of a project are usually based on a range of possible values (e.g., the design phase will take three to four months) and gradually become more specific as the project moves forward (e.g., the design phase will be completed on March 22).

The numbers used to calculate these estimates can come from several sources. They can be provided with the methodology that is used, taken from projects with similar tasks and technologies, or provided by experienced developers. Generally speaking, the numbers should be conservative. A good practice is to keep track of the actual time and effort values during the SDLC so that numbers can be refined along the way, and the next project can benefit from real data. One of the greatest strengths of systems consulting firms is the past experience that they offer to a project; they have estimates and methodologies that have been developed and honed over time and applied to hundreds of projects.

There are two basic ways to estimate the time required to build a system. The simplest method uses the amount of time spent in the planning phase to predict the time required for the entire project. The idea is that a simple project will require little planning, and a complex project will require more planning; so using the amount of time spent in the planning phase is a reasonable way to estimate overall project time requirements.

With this approach, you take the time spent in (or estimated for) the planning phase and use industry standard percentages (or percentages from the organization’s own experiences) to calculate estimates for the other SDLC phases. Industry standards suggest that a “typical” business application system spends 15% of its effort in the planning phase, 20% in the analysis phase, 35% in the design phase, and 30% in the implementation phase. This would suggest that if a project takes four months in the planning phase, then the rest of the project likely will take a total of 22.66 person-months ($4 / .15 = 22.66$). These same industry percentages are then used to estimate the amount of time in each phase (Figure 2-10). The obvious limitation of this approach is that it can be difficult to take into account the specifics of your individual project, which may be simpler or more difficult than the “typical” project.

A more precise approach to estimation is called the function point approach. This approach is a more complex—and, it is hoped, more reliable—way of estimating time and effort for a project. The details of the function point approach are explained in Appendix 2A.

<table>
<thead>
<tr>
<th></th>
<th>Planning</th>
<th>Analysis</th>
<th>Design</th>
<th>Implementation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Typical industry standards for business applications</td>
<td>15%</td>
<td>20%</td>
<td>35%</td>
<td>30%</td>
</tr>
<tr>
<td>Estimates based on actual figures for first stage of SDLC</td>
<td>Actual: 4 person-months</td>
<td>Estimated: 5.33 person-months</td>
<td>Estimated: 9.33 person-months</td>
<td>Estimated: 8 person-months</td>
</tr>
</tbody>
</table>

SDLC = systems development life cycle.

**FIGURE 2-10** Estimating Project Time Using Industry Standards
Developing the Work Plan

Once a project manager has a general idea of the size and approximate schedule for the project, he or she creates a *work plan*, which is a dynamic schedule that records and keeps track of all of the tasks that need to be accomplished over the course of the project. The project manager first must assemble important details about each task to be completed. Figure 2-11 shows the type of task information needed, including when it needs to be completed, the person assigned to do the work, and any deliverables that will result. The level of detail and the amount of information captured by the work plan depend on the needs of the project (and the detail usually increases as the project progresses). Usually, the work plan is the main component of the project management software that we mentioned earlier.

To create a work plan, the project manager identifies the tasks that need to be accomplished and determines how long each one will take. Then the tasks are organized within a work breakdown structure.

### Identify Tasks

Remember that the overall objectives for the system were recorded on the system request, and the project manager’s job is to identify all the tasks that will be needed to accomplish those objectives. This is a daunting task, certainly. The methodology that was selected by the project manager should be a valuable resource, however. The methodology that seems most appropriate for the project provides a list of steps and deliverables.

A project manager can take the methodology, select the steps and deliverables that apply to the current project, and add them to the work plan. If an existing methodology is not available within the organization, methodologies can be purchased from consultants or vendors, or books like this textbook can serve as guidance. Using an existing methodology is the most popular way to create a work plan, because most organizations have a methodology that they use for projects.

If a project manager prefers to begin from scratch, he or she can use a structured, top-down approach whereby high-level tasks are defined first and then broken down into subtasks. Each step is then broken down in turn and numbered in a hierarchical fashion. A list of tasks hierarchically numbered in this way is called a *work breakdown structure*, and it is the backbone of the project.
workplan. Figure 2-12 shows a portion of a work breakdown structure for the design phase of an actual data warehouse development project. Each of the main tasks focuses on one of the required design deliverables. Within each task, there are subtasks listed that detail the activities required to complete the main task.

The work breakdown structure can be organized in one of two ways: by SDLC phase or by product. For example, if a firm decided that it needed to develop a Web site, the firm could create a work breakdown structure based on the SDLC phases: planning, analysis, design, and implementation. In this case, a typical task that would take place during planning would be feasibility analysis. This task would be broken down into the different types of feasibility analysis: technical, economic, and organizational. Each of these would be further broken down into a series of subtasks. Alternatively, the firm could organize the work plan along the lines of the different products to be developed. In the case of a Web site, for example, the products could include applets, application servers, database servers, the various sets of Web pages, a site map, and so on. Each of these products could be decomposed into the different tasks associated with the phases of the SDLC. With either approach, once the overall structure is determined, tasks are identified and included in the work breakdown structure of the work plan.
The number of tasks and level of detail depend on the complexity and size of the project. The larger the project, the more important it becomes to define tasks in detail so that essential steps are not overlooked.

**The Project Work Plan**  The project work plan is the mechanism used to manage the tasks that are listed in the work breakdown structure. It is the project manager’s primary tool for managing the project. Using it, the project manager can tell whether the project is ahead of or behind schedule, how well the project was estimated, and what changes need to be made to meet the project deadline.

Basically, the work plan is a table that lists all of the tasks in the work breakdown structure, along with important task information such as the people who are assigned to perform the tasks, the actual hours that the tasks took, and the variances between estimated and actual completion times. (See Figure 2-13). At a minimum, the information should include the duration of the task, the current statuses of the tasks (i.e., open, complete), and the *task dependencies*, which occur when one task cannot be performed until another task is completed. For example, Figure 2-13 shows that task 1.2 and task 1.3 cannot begin until task 1.1 is completed. Key *milestones*, or important dates, are also identified on the work plan. Presentations to the approval committee, the start of end-user training, a company retreat, and the due date of the system prototype are the types of milestones that may be important to track.

### STAFFING THE PROJECT

Staffing the project includes determining how many people should be assigned to the project, matching people’s skills with the needs of the project, motivating them to meet the project’s objectives, and minimizing project team conflict that will occur over time. The deliverable for this part of project management is a staffing plan, which describes the number and kinds of people who will work on the project, the overall reporting structure, and the project charter, which describes the project’s objectives and rules.

**Staffing Plan**  The first step to staffing is determining the average number of staff needed for the project. To calculate this figure, divide the total person-months of effort by the optimal schedule. So to complete a 40 person-month project in 10 months, a team should have an average of four full-time staff members, although this may change over time as different specialists enter and leave the team (e.g., business analysts, programmers, technical writers).

Many times, the temptation is to assign more staff to a project to shorten the project’s length, but this is not a wise move. Adding staff resources does not translate into increased productivity; staff size and productivity share a disproportionate relationship, mainly because a large number of staff members is more difficult to coordinate. The more a team grows, the more difficult it becomes to manage. Imagine how easy it is to work on a two-person project team: the team members share a single line of communication. But adding two people increases the number of communication lines to six, and greater increases lead to more dramatic gains in communication complexity. Figure 2-14 and Your Turn 2-4 illustrate the impact of adding team members to a project team.

One way to reduce efficiency losses on teams is to understand the complexity that is created in numbers and to build in a *reporting structure* that tempers its effects.
<table>
<thead>
<tr>
<th>Task ID</th>
<th>Task Name</th>
<th>Assigned To</th>
<th>Start Date</th>
<th>Finish Date</th>
<th>Start Date</th>
<th>Finish Date</th>
<th>Duration (days)</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>Develop database design</td>
<td>Megan</td>
<td>12/1/13</td>
<td>12/3/13</td>
<td>12/14/13</td>
<td>12/26/13</td>
<td>9</td>
<td>Open</td>
</tr>
<tr>
<td>1.1.1</td>
<td>Staging database design</td>
<td>Megan</td>
<td>12/1/13</td>
<td>12/3/13</td>
<td>12/14/13</td>
<td>12/26/13</td>
<td>9</td>
<td>Open</td>
</tr>
<tr>
<td>1.2</td>
<td>Develop rejects-handling design</td>
<td>Megan</td>
<td>12/1/13</td>
<td>12/3/13</td>
<td>12/14/13</td>
<td>12/26/13</td>
<td>9</td>
<td>Open</td>
</tr>
<tr>
<td>1.2.1</td>
<td>Rejects-handling engine design</td>
<td>Megan</td>
<td>12/1/13</td>
<td>12/3/13</td>
<td>12/14/13</td>
<td>12/26/13</td>
<td>9</td>
<td>Open</td>
</tr>
<tr>
<td>1.3</td>
<td>Develop OLAP design</td>
<td>Joachim</td>
<td>12/1/13</td>
<td>12/3/13</td>
<td>12/14/13</td>
<td>12/26/13</td>
<td>9</td>
<td>Open</td>
</tr>
<tr>
<td>1.3.1</td>
<td>Universe design</td>
<td>Joachim</td>
<td>12/1/13</td>
<td>12/3/13</td>
<td>12/14/13</td>
<td>12/26/13</td>
<td>9</td>
<td>Open</td>
</tr>
<tr>
<td>1.4</td>
<td>Develop OLAP design part 1</td>
<td>Kevin</td>
<td>12/1/13</td>
<td>12/3/13</td>
<td>12/14/13</td>
<td>12/26/13</td>
<td>8</td>
<td>Open</td>
</tr>
<tr>
<td>1.4.1</td>
<td>High-priority reports design</td>
<td>Kevin</td>
<td>12/1/13</td>
<td>12/3/13</td>
<td>12/14/13</td>
<td>12/26/13</td>
<td>8</td>
<td>Open</td>
</tr>
<tr>
<td>1.5</td>
<td>Develop application design</td>
<td>Tomas</td>
<td>12/1/13</td>
<td>12/3/13</td>
<td>12/14/13</td>
<td>12/26/13</td>
<td>9</td>
<td>Open</td>
</tr>
<tr>
<td>1.5.1</td>
<td>Group consolidation and corporate reporting design</td>
<td>Tomas</td>
<td>12/1/13</td>
<td>12/3/13</td>
<td>12/14/13</td>
<td>12/26/13</td>
<td>9</td>
<td>Open</td>
</tr>
<tr>
<td>Task ID</td>
<td>Task Name</td>
<td>Assigned To</td>
<td>Estimated</td>
<td>Actual</td>
<td>Dependency</td>
<td>Status</td>
<td></td>
<td></td>
</tr>
<tr>
<td>--------</td>
<td>----------------------------------------------------</td>
<td>-------------</td>
<td>-----------</td>
<td>--------</td>
<td>------------</td>
<td>--------</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Duration</td>
<td>Start Date</td>
<td>Finish Date</td>
<td>Start Date</td>
<td>Finish Date</td>
<td>Duration Variance</td>
</tr>
<tr>
<td>1.6</td>
<td>Extract, transform, load (ETL) design document</td>
<td>Joachim</td>
<td>2</td>
<td>Thu 12/27/13</td>
<td>Fri 12/28/13</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.6.1</td>
<td>Data export utility design</td>
<td>Joachim</td>
<td>2</td>
<td>Thu 12/27/13</td>
<td>Fri 12/28/13</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.7</td>
<td>Application design document</td>
<td>Mei-ling</td>
<td>25</td>
<td>Fri 11/16/13</td>
<td>Fri 12/21/13</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.7.1</td>
<td>Web entry application UI design</td>
<td>Mei-ling</td>
<td>25</td>
<td>Fri 11/16/13</td>
<td>Fri 12/21/13</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.7.2</td>
<td>Web entry application UI design sign-off</td>
<td>Mei-ling</td>
<td>1</td>
<td>Fri 12/30/13</td>
<td>Fri 12/30/13</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.7.3</td>
<td>Web entry forms and database model validation</td>
<td>Kevin</td>
<td>11</td>
<td>Wed 11/21/13</td>
<td>Wed 12/5/13</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.8</td>
<td>Functional requirements document</td>
<td>Chantelle</td>
<td>9</td>
<td>Mon 12/10/13</td>
<td>Thu 12/20/13</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.8.1</td>
<td>Application design</td>
<td>Chantelle</td>
<td>9</td>
<td>Mon 12/10/13</td>
<td>Thu 12/20/13</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.8.1.1</td>
<td>User authentication</td>
<td>Chantelle</td>
<td>4</td>
<td>Mon 12/10/13</td>
<td>Thu 12/13/13</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.8.1.2</td>
<td>Call logging</td>
<td>Chantelle</td>
<td>2</td>
<td>Fri 12/14/13</td>
<td>Mon 12/17/13</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.8.1.3</td>
<td>Search</td>
<td>Chantelle</td>
<td>3</td>
<td>Tue 12/18/13</td>
<td>Thu 12/20/13</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(Thanks to Priya Padmanabhan for suggesting this example.)
The rule of thumb is to keep team sizes under 8 to 10 people; therefore, if more people are needed, create subteams. In this way, the project manager can keep the communication effective within small teams, which in turn communicate to a contact at a higher level in the project.

After the project manager understands how many people are needed for the project, he or she creates a staffing plan that lists the roles that are required for the project and the proposed reporting structure for the project. Typically, a project will have one project manager who oversees the overall progress of the development effort, with the core of the team composed of the various types of analysts described in Chapter 1. A functional lead usually is assigned to manage a group of analysts, and a technical lead oversees the progress of a group of programmers and more technical staff members.

There are many structures for project teams; Figure 2-15 illustrates one possible configuration of a project team. After the roles are defined and the structure is in place, the project manager needs to think about which people can fill each role. Often, one person fills more than one role on a project team.

When you make assignments, remember that people have technical skills and interpersonal skills, and both are important on a project. Technical skills are useful for working with technical tasks (e.g., programming in Java) and in trying to understand the various roles that technology plays in the particular project (e.g., how a Web server should be configured on the basis of a projected number of hits from customers).

Interpersonal skills, on the other hand, include interpersonal and communication abilities that are used when dealing with business users, senior management...
executives, and other members of the project team. They are particularly critical for performing the requirements-gathering activities and when addressing organizational feasibility issues. Each project will require unique technical and interpersonal skills. For example, a Web-based project may require Internet experience or Java programming knowledge, or a highly controversial project may need analysts who are particularly adept at managing political or volatile situations.

Ideally, project roles are filled with people who have the right skills for the job; however, the people who fit the roles best may not be available; they may be working on other projects, or they may not exist in the company. Therefore, assigning project team members really is a combination of finding people with the appropriate skill sets and finding people who are available. When the skills of the available project team members do not match those actually required by the project, the project manager has several options to improve the situation. First, people can be pulled off other projects, and resources can be shuffled around. This is the most disruptive approach from the organization’s perspective. Another approach is to use outside help—such as a consultant or contractor—to train team members and start them off on the right foot. Training classes are usually available for both technical and interpersonal instruction, if time is available. Mentoring may also be an option; a project team member can be sent to work on another similar project so that he or she can return with skills to apply to the current job.

Motivation Assigning people to tasks isn’t enough; project managers need to motivate the people to make the project a success. Motivation has been found to be the number-one influence on people’s performance, but determining how to motivate the team can be quite difficult. You may think that good project managers motivate their staff by rewarding them with money and bonuses, but most project managers agree that this is the last thing that should be done. The more often you reward team members with money, the more they expect it—and most times monetary motivation won’t work.

Assuming that team members are paid a fair salary, technical employees on project teams are much more motivated by recognition, achievement, the work

---

itself, responsibility, advancement, and the chance to learn new skills.\footnote{F. H. Hertzberg, “One More Time: How Do You Motivate Employees?” \textit{Harvard Business Review}, 1968, January–February.} If you feel that you need to give some kind of reward for motivational purposes, try a pizza or free dinner, or even a kind letter or award. These often have much more effective results. Figure 2-16 lists some other motivational don’ts that you should avoid to ensure that motivation on the project is as high as possible.

### Handling Conflict

The third component of staffing is organizing the project to minimize conflict among group members. Group cohesiveness (the attraction that members feel to the group and to other members) contributes more to productivity than do project members’ individual capabilities or experiences.\footnote{B. Lakhanpal, “Understanding the Factors Influencing the Performance of Software Development Groups: An Exploratory Group-Level Analysis,” \textit{Information and Software Technology}, 1993, 35(8):468–473.} Clearly defining the roles on the project and holding team members accountable for their tasks is a good way to begin mitigating potential conflict on a project. Some project managers develop a project charter that lists the project’s norms and ground rules. For example, the charter may describe when the project team should be at work, when staff meetings will be held, how the group will communicate with each other, and the procedures for updating the work plan as tasks are completed. Figure 2-17 lists additional techniques that can be used at the start of a project to keep conflict to a minimum.

### Coordinating Project Activities

Like all project management responsibilities, the act of coordinating project activities continues throughout the entire project until a system is delivered to the project sponsor and end users. This step includes putting efficient development practices in

<table>
<thead>
<tr>
<th>Don’ts</th>
<th>Reasons</th>
</tr>
</thead>
<tbody>
<tr>
<td>Assign unrealistic deadlines</td>
<td>Few people will work hard if they realize that a deadline is impossible to meet.</td>
</tr>
<tr>
<td>Ignore good efforts</td>
<td>People will work harder if they feel that their work is appreciated. Often, all it takes is public praise for a job well done.</td>
</tr>
<tr>
<td>Create a low-quality product</td>
<td>Few people can be proud of working on a project that is of low quality.</td>
</tr>
<tr>
<td>Give everyone on the project a raise</td>
<td>If everyone is given the same reward, then high-quality people will believe that mediocrity is rewarded—and they will resent it.</td>
</tr>
<tr>
<td>Make an important decision without the team’s input</td>
<td>Buy-in is very important. If the project manager needs to make a decision that greatly affects the members of her team, she should involve them in the decision-making process.</td>
</tr>
<tr>
<td>Maintain poor working conditions</td>
<td>A project team needs a good working environment, or motivation will go down the tubes. This includes lighting, desk space, technology, privacy from interruptions, and reference resources.</td>
</tr>
</tbody>
</table>

Clearly define plans for the project.
Make sure the team understands how the project is important to the organization.
Develop detailed operating procedures and communicate these to the team members.
Develop a project charter.
Develop schedule commitments ahead of time.
Forecast other priorities and their possible impact on project.


place and mitigating risk. These activities occur over the course of the entire SDLC, but it is at this point in the project that the project manager needs to put them in place. Ultimately, these activities ensure that the project stays on track and that the chance of failure is kept at a minimum. The rest of this section will describe each of these activities in more detail.

**CASE Tools**  
*Computer-aided software engineering (CASE)* is a category of software that automates all or part of the development process. Some CASE software packages are primarily used during the analysis phase to create integrated diagrams of the system and to store information regarding the system components (often called *upper CASE*), whereas others are design-phase tools that create the diagrams and then generate code for database tables and system functionality (often called *lower CASE*). *Integrated CASE*, or I-CASE, contains functionality found in both upper-CASE and lower-CASE tools in that it supports tasks that happen throughout the SDLC. CASE comes in a wide assortment of flavors in terms of complexity and functionality, and there are many good programs available in the marketplace, such as the Visible Analyst Workbench, Oracle Designer, Rational Rose, and the Logic Works suite.

The benefits of using CASE are numerous. With CASE tools, tasks are much faster to complete and alter; development information is centralized; and information is illustrated through diagrams, which typically are easier to understand. Potentially, CASE can reduce maintenance costs, improve software quality, and enforce discipline; and some project teams even use CASE to assess the magnitude of changes to the project.

Of course, like anything else, CASE should not be considered a silver bullet for project development. The advanced CASE tools are complex applications that require significant training and experience to achieve real benefits. Often, CASE serves only as a glorified diagramming tool that supports the practices described in Chapter 5 (process modeling) and Chapter 6 (data modeling). Our experience has shown that CASE is a helpful way to support the communication and sharing of project diagrams and technical specifications—as long as it is used by trained developers who have applied CASE on past projects.

The central component of any CASE tool is the *CASE repository*, otherwise known as the information repository or data dictionary. The CASE repository stores the diagrams and other project information, such as screen and report designs, and it keeps track of how the diagrams fit together. For example, most CASE tools will warn you if you place a field on a screen design that doesn’t exist in your data model. As the project evolves, project team members perform their tasks by using CASE. As you read through the textbook, we will indicate when and how the CASE tool can be used so that you can see how CASE supports the project tasks.
Standards  Members of a project team need to work together, and most project management software and CASE tools provide access privileges to everyone working on the system. When people work together, however, things can get pretty confusing. To make matters worse, people sometimes get reassigned in the middle of a project. It is important that their project knowledge does not leave with them and that their replacements can get up to speed quickly.

Standards are created to ensure that team members are performing tasks in the same way and following the same procedures. Standards can range from formal rules for naming files to forms that must be completed when goals are reached to programming guidelines. See Figure 2-18 for some examples of the types of standards that a project may include. When a team forms standards and then follows them, the project can be completed faster because task coordination becomes less complex.

Standards work best when they are created at the beginning of each major phase of the project and well communicated to the entire project team. As the team moves forward, new standards are added when necessary. Some standards (e.g., file-naming conventions, status reporting) are applied to the entire SDLC, whereas others (e.g., programming guidelines) are appropriate only for certain tasks.

Documentation  Another technique that project teams put in place during the planning phase is good documentation, which includes detailed information about the tasks of the SDLC. Often, the documentation is stored in project binder(s) that contain all the deliverables and all the internal communication that takes place—the history of the project.

A poor project management practice is permitting the project team to wait until the last minute to create documentation. This typically leads to an undocumented system that no one understands. In fact, many problems that companies had in updating their systems to handle the year-2000 crisis were the result of the lack of documentation. Good project teams learn to document the system’s history as it evolves, while the details are still fresh in their memory.

A simple way to set up your documentation is to get some binders and use dividers to separate content according to the major phases of the project. An additional divider should contain internal communication, such as the minutes from status meetings, written standards, letters to and from the business users, and a dictionary of relevant business terms. Then, as the project moves forward, place the deliverables from each task into the project binder with descriptions so that someone outside of the project will be able to understand it, and keep a table of contents up to date with the content that is added. Documentation takes time up front, but it is a good investment that will pay off in the long run.

Select a computer-aided software engineering (CASE) tool—either one that you will use for class, a program that you own, or a tool that you can examine over the Web. Create a list of the capabilities that are offered by the CASE tool.

Question: Would you classify the CASE as upper CASE, lower CASE, or integrated CASE (I-CASE)? Why?
The science (or art) of project management is in making trade-offs among three important concepts: the size of the system (in terms of what it does), the time to complete the project (when the project will be finished), and the cost of the project. Think of these three things as interdependent levers that the project manager controls throughout the SDLC. Whenever one lever is pulled, the other two levers are affected in some way. For example, if a project manager needs to readjust a deadline to an earlier date, then the only solution is to decrease the size of the system (by eliminating some of its functions) or to increase costs by adding more people or having team members work overtime. Often, a project manager will have to work with the project sponsor to change the goals of the project, such as developing a system with less functionality or extending the deadline for the final system, so that the project has reasonable goals that can be met.

Therefore, in the beginning of the project, the manager needs to estimate each of these levers and then continuously assess how to roll out the project in a way that meets the organization’s needs.

### Types of Standards

<table>
<thead>
<tr>
<th>Types of Standards</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Documentation standards</td>
<td>The date and project name should appear as a header on all documentation.</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Coding standards</td>
<td>All modules of code should include a header that lists the programmer, last date of update, and a short description of the purpose of the code.</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Procedural standards</td>
<td>Record actual task progress in the work plan every Monday morning by 10 A.M.</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Specification requirement standards</td>
<td>Name of program to be created</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>User interface design standards</td>
<td>Labels will appear in boldface text, left-justified, and followed by a colon.</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### MANAGING AND CONTROLLING THE PROJECT

The science (or art) of project management is in making trade-offs among three important concepts: the size of the system (in terms of what it does), the time to complete the project (when the project will be finished), and the cost of the project. Think of these three things as interdependent levers that the project manager controls throughout the SDLC. Whenever one lever is pulled, the other two levers are affected in some way. For example, if a project manager needs to readjust a deadline to an earlier date, then the only solution is to decrease the size of the system (by eliminating some of its functions) or to increase costs by adding more people or having team members work overtime. Often, a project manager will have to work with the project sponsor to change the goals of the project, such as developing a system with less functionality or extending the deadline for the final system, so that the project has reasonable goals that can be met.

Therefore, in the beginning of the project, the manager needs to estimate each of these levers and then continuously assess how to roll out the project in a way that meets the organization’s needs.
Once the project begins, the project manager monitors the progress of the team on the project tasks. As the project team members make periodic status reports, the project manager updates the project work plan. As discussed in Appendix 2B, the Gantt chart and PERT chart are valuable tools for the project manager to use to evaluate project progress and, if necessary, redirect resources. As the project proceeds, it may be necessary for the project manager to revise the original estimates made for the project. In addition, the manager must be on the watch for increases in project scope, which can make completing the project on time and under budget very difficult. Finally, the project manager should constantly assess the risk profile of the project and take steps to manage those risks.

Refining Estimates

The estimates that are produced during the planning phase will need to be refined as the project progresses. This does not necessarily mean that estimates were poorly done at the start of the project; it is virtually impossible to develop an exact assessment of the project’s schedule before the analysis and design phases are conducted. A project manager should expect to be satisfied with broad ranges of estimates that become more and more specific as the project’s product becomes better defined.

In the planning phase, when a system is first requested, the project sponsor and project manager attempt to predict how long the SDLC will take, how much it will cost, and what the system will ultimately do when it is delivered (i.e., its functionality). However, the estimates are based on very little knowledge of the system. As the project moves into the analysis phase, more information is gathered, the system concept is developed, and the estimates become even more accurate and precise. As the system moves closer to completion, the accuracy and precision increase until the final system is delivered.

2-G Trade-Offs

I was once on a project to develop a system that should have taken a year to build. Instead, the business need demanded that the system be ready within 5 months—impossible!

On the first day of the project, the project manager drew a triangle on a white board to illustrate some tradeoffs that he expected to occur over the course of the project. The corners of the triangle were labeled Functionality, Time, and Money. The manager explained, “We have too little time. We have an unlimited budget. We will not be measured by the bells and whistles that this system contains. So over the next several weeks, I want you as developers to keep this triangle in mind and do everything it takes to meet this 5-month deadline.”

At the end of the 5 months, the project was delivered on time; however, the project was incredibly over budget, and the final product was “thrown away” after it was used because it was unfit for regular usage. Remarkably, the business users felt that the project was very successful because it met the very specific business needs for which it was built. They believed that the trade-offs that were made were worthwhile. Barbara Wixom

**Questions:**

1. What are the risks in stressing only one corner of the triangle?
2. How would you have managed this project? Can you think of another approach that might have been more effective?
According to one of the leading experts in software development, a well-done project plan (prepared at the end of the planning phase) has a 100% margin of error for project cost and a 25% margin of error for schedule time. In other words, if a carefully done project plan estimates that a project will cost $100,000 and take 20 weeks, the project will actually cost between $0 and $200,000 and take between 15 and 25 weeks. Figure 2-19 presents typical margins of error for other stages in the project. It is important to note that these margins of error apply only to well-done plans; a plan developed without much care has a much greater margin of error.

What happens if you overshoot an estimate (e.g., the analysis phase ends up lasting two weeks longer than expected)? There are a number of ways to adjust future estimates. If the project team finishes a step ahead of schedule, most project managers shift the deadlines sooner by the same amount but do not adjust the promised completion date. The challenge, however, occurs when the project team is late in meeting a scheduled date. Three possible responses to missed schedule dates are presented in Figure 2-20. We recommend that if an estimate proves too optimistic early in the project, do not expect to make up for lost time—very few projects end up working this way. Instead, change your future estimates to include an increase similar to the one that was experienced. For example, if the first phase was completed 10% over schedule, increase the rest of your estimates by 10%.

**Managing Scope**

You may assume that your project will be safe from scheduling problems because you carefully estimated and planned your project up front. However, the most common reason for schedule and cost overruns occurs after the project is underway—scope creep.

Scope creep happens when new requirements are added to the project after the original project scope was defined and “frozen.” It can happen for many reasons: Users may suddenly understand the potential of the new system and realize new functionality that would be useful; developers may discover interesting capabilities to which they become very attached; a senior manager may decide to let this system support a new strategy that was developed at a recent board meeting.

---

Unfortunately, after the project begins, it becomes increasingly difficult to address changing requirements. The ramifications of change become more extensive, the focus is removed from original goals, and there is at least some impact on cost and schedule. Therefore, the project manager must actively work to keep the project tight and focused.

The keys are to identify the requirements as well as possible in the beginning of the project and to apply analysis techniques effectively. For example, if needs are fuzzy at the project’s onset, a combination of intensive meetings with the users and prototyping could be used so that users “experience” the requirements and better visualize how the system could support their needs. In fact, the use of meetings and prototyping has been found to reduce scope creep to less than 5% on a typical project.

Of course, some requirements may be missed no matter what precautions you take, but several practices can help to control additions to the task list. First, the project manager should allow only absolutely necessary requirements to be added after the project begins. Even at that point, members of the project team should carefully assess the ramifications of the addition and present the assessment back to the users. For example, it may require two more person-months of work to create a newly defined report, which would throw off the entire project deadline by several weeks. Any change that is implemented should be carefully tracked so that an audit trail exists to measure the change’s impact.

Sometimes, changes cannot be incorporated into the present system even though they truly would be beneficial. In this case, these additions to scope should
be recorded as future enhancements to the system. The project manager can offer to provide functionality in future releases of the system, thus getting around telling someone no.

**Timeboxing**

Up until now, we have described projects that are task oriented. In other words, we have described projects that have a schedule that is driven by the tasks that need to be accomplished, so the greater number of tasks and requirements, the longer the project will take. Some companies have little patience for development projects that take a long time, and these companies take a time-oriented approach that places meeting a deadline above delivering functionality.

Think about your use of word processing software. For 80% of the time, you probably use only 20% of the features, such as the spelling checker, boldfacing, and cutting and pasting. Other features, such as document merging and creation of mailing labels, may be nice to have, but they are not a part of your day-to-day needs. The same goes for other software applications; most users rely on only a small subset of their capabilities. Ironically, most developers agree that, typically, 75% of a system can be provided relatively quickly, with the remaining 25% of the functionality demanding most of the time.

To resolve this incongruency, a technique called *timeboxing* has become quite popular, especially when rapid application development (RAD) methodologies are used. This technique sets a fixed deadline for a project and delivers the system by that deadline no matter what, even if functionality needs to be reduced. Timeboxing ensures that project teams don’t get hung up on the final “finishing touches” that can drag out indefinitely, and it satisfies the business by providing a product within a relatively fast time frame.

There are several steps to implementing timeboxing on a project (Figure 2-21). First, set the date of delivery for the proposed goals. The deadline should not be impossible to meet, so it is best to let the project team determine a realistic due date. Next, build the core of the system to be delivered; you will find that timeboxing helps create a sense of urgency and helps keep the focus on the most important features. Because the schedule is absolutely fixed, functionality that cannot be completed needs to be postponed. It helps if the team prioritizes a list of features beforehand to keep track of what functionality the users absolutely need. Quality cannot be compromised, regardless of other constraints, so it is important that the time allocated to activities is not shortened unless the requirements are changed (e.g., don’t reduce the time allocated to testing without reducing features). At the end of the period, a high-quality system is delivered. Likely, future iterations will be needed to make changes and enhancements, and the timeboxing approach can be used once again.

---

**FIGURE 2-21**

Steps for Timeboxing

1. Set the date for system delivery.
2. Prioritize the functionality that needs to be included in the system.
3. Build the core of the system (the functionality ranked as most important).
4. Postpone functionality that cannot be provided within the time frame.
5. Deliver the system with core functionality.
6. Repeat steps 3 through 5, to add refinements and enhancements.
Managing Risk

One final facet of project management is risk management, the process of assessing and addressing the risks that are associated with developing a project. Many things can cause risks: weak personnel, scope creep, poor design, and overly optimistic estimates. The project team must be aware of potential risks so that problems can be avoided or controlled well ahead of time.

Typically, project teams create a risk assessment, or a document that tracks potential risks along with an evaluation of the likelihood of the risk and its potential impact on the project (Figure 2-22). A paragraph or two is included that explains potential ways that the risk can be addressed. There are many options: A risk could be publicized, avoided, or even eliminated by dealing with its root cause. For example, imagine that a project team plans to use new technology, but its members have identified a risk in the fact that its members do not have the right technical skills. They believe that tasks may take much longer to perform because of a high learning curve. One plan of attack could be to eliminate the root cause of the risk—the lack of technical experience by team members—by finding time and resources that are needed to provide proper training to the team.
Most project managers keep abreast of potential risks, even prioritizing them according to their magnitude and importance. Over time, the list of risks will change as some items are removed and others surface. The best project managers, however, work hard to keep risks from having an impact on the schedule and costs associated with the project.

RISK ASSESSMENT

**RISK #1:**

The development of this system likely will be slowed considerably because project team members have not programmed in Java prior to this project.

**Likelihood of risk:**

High probability of risk

**Potential impact on the project:**

This risk likely will increase the time to complete programming tasks by 50%.

**Ways to address this risk:**

It is very important that time and resources are allocated to up-front training in Java for the programmers who are used for this project. Adequate training will reduce the initial learning curve for Java when programming begins. Additionally, outside Java expertise should be brought in for at least some part of the early programming tasks. This person should be used to provide experiential knowledge to the project team so that Java-related issues (of which novice Java programmers would be unaware) are overcome.

**RISK #2:**

etc…. 
APPLYING THE CONCEPTS AT TUNE SOURCE

Jason Wells was very excited about managing the Digital Music Download project at Tune Source, but he realized that his project team should move rapidly to deliver at least some parts of the system because of the company’s desire to bring the application to market as quickly as possible. Therefore, he decided that the project should follow a RAD iterative development methodology, combined with the time-boxing technique. In this way, he could be sure that some version of the system would be operational within several months, even if the final system was delivered at a later date.

Jason knew that Carly Edwards and Tune Source’s top managers wanted at least general ranges for a product delivery date. He expected to spend three weeks planning the project. Therefore, using industry standard percentages, he estimated that the entire project should be complete in 20 weeks (3 weeks/15%). Recall from Figure 2-10 that the planning phase typically takes 15% of the entire project. Jason’s initial plan was to develop the basic music download purchase capability in the first version of the system. The second version will add the subscription purchase capability, and the third version will add the gift card purchase capability.

2-1 AVOIDING CLASSIC PLANNING MISTAKES

As Seattle University’s David Umphress has pointed out, watching most organizations develop systems is like watching reruns of Gilligan’s Island. At the beginning of each episode, someone comes up with a cockamamie scheme to get off the island that seems to work for a while, but something goes wrong and the castaways find themselves right back where they started—stuck on the island. Similarly, most companies start new projects with grand ideas that seem to work, only to make a classic mistake and deliver the project behind schedule, over budget, or both. Here we summarize four classic mistakes in the planning and project management aspects of the project and discuss how to avoid them:

1. Overly optimistic schedule: Wishful thinking can lead to an overly optimistic schedule that causes analysis and design to be cut short (missing key requirements) and puts intense pressure on the programmers, who produce poor code (full of bugs).
   Solution: Don’t inflate time estimates; instead, explicitly schedule slack time at the end of each phase to account for the variability in estimates, using the margins of error from Figure 2-19.

2. Failing to monitor the schedule: If the team does not regularly report progress, no one knows if the project is on schedule.
   Solution: Require team members to honestly report progress (or the lack of progress) every week. There is no penalty for reporting a lack of progress, but there are immediate sanctions for a misleading report.

3. Failing to update the schedule: When a part of the schedule falls behind (e.g., information gathering uses all of the slack in item 1 above plus 2 weeks), a project team often thinks it can make up the time later by working faster. It can’t. This is an early warning that the entire schedule is too optimistic.
   Solution: Immediately revise the schedule and inform the project sponsor of the new end date or use timeboxing to reduce functionality or to move it into future versions.

4. Adding people to a late project: When a project misses a schedule, the temptation is to add more people to speed it up. This makes the project take longer because it increases coordination problems and requires staff to take time to explain what has already been done.
   Solution: Revise the schedule, use timeboxing, throw away bug-filled code, and add people only to work on an isolated part of the project.

With a general time frame determined, Jason began to identify the tasks that would be needed to complete the system. He used a RAD methodology that Tune Source had in-house, and he borrowed its high-level phases (e.g., analysis) and the major tasks associated with them (e.g., create requirements definition, gather requirements, analyze current system). These were recorded in a work plan created in Microsoft Project. Jason expected to define the steps in much more detail at the beginning of each phase (Figure 2-23).

**Staffing the Project**

Jason next turned to the task of how to staff his project. According to his earlier estimates, it appeared that about three people would be needed to deliver the system.

First, he created a list of the various roles that he needed to fill. He thought he would need several analysts to work with the analysis and design of the Digital Music Download system, as well as an infrastructure analyst to manage the integration of the system with Tune Source’s existing technical environment. Jason also needed people who had good programming skills and who could be responsible for ultimately implementing the system. Kenji, Ming, and Maria are three analysts with strong technical and interpersonal skills (although Kenji is less balanced, having greater technical than interpersonal abilities), and Jason believed that they were available to bring onto this project. He wasn’t certain whether they had experience with the actual Web technology that would be used on the project, but he decided to rely on vendor training or an external consultant to build those skills later when they were needed. Because the project was so small, Jason envisioned all of the team members reporting to him because he would be serving as the project’s manager.

Jason created a staffing plan that captured this information, and he included a special incentive structure in the plan (Figure 2-24). Rapid implementation was very important to the project’s success, so he decided to offer a day off to the team members who contributed to meeting that date. He hoped that this incentive would motivate the team to work very hard. Jason also planned to budget money for pizza and sodas for times when the team worked long hours.

Before he left for the day, Jason drafted a project charter, to be fine-tuned after the team got together for its kick-off meeting (i.e., the first time the project team gets together). The charter listed several norms that Jason wanted to put in place from the start to eliminate any misunderstanding or problems that could come up otherwise (Figure 2-25).

**Coordinating Project Activities**

Jason wanted the Digital Music Download project to be well coordinated, so he immediately put several practices in place to support his responsibilities. First, he acquired the CASE tool used at Tune Source and set up the product so that it could be used for the analysis-phase tasks (e.g., drawing the data flow diagrams). The team members would likely start creating diagrams and defining components of the system fairly early on. He pulled out some standards that he uses on all development projects and made a note to review them with his project team at the kick-off meeting for the system. He also had his assistant set up binders for the project deliverables that would start rolling in. Already, he was able to include the system request, feasibility analysis, initial work plan, staffing plan, project charter, standards list, and risk assessment.
### FIGURE 2-23
Gantt Chart

<table>
<thead>
<tr>
<th>ID</th>
<th>Task Name</th>
<th>Duration</th>
<th>Start</th>
<th>Finish</th>
<th>Predecessors</th>
<th>Resource Names</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Overall analysis</td>
<td>10 days</td>
<td>Mon 2/4/13</td>
<td>Fri 2/15/13</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Identify High-Level Requirements</td>
<td>6 days</td>
<td>Mon 2/4/13</td>
<td>Mon 2/11/13</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>JAD session</td>
<td>4 days</td>
<td>Mon 2/4/13</td>
<td>Thu 2/7/13</td>
<td>3</td>
<td>Jason, Carly, Ming</td>
</tr>
<tr>
<td>4</td>
<td>Informal benchmarking</td>
<td>2 days</td>
<td>Fri 2/8/13</td>
<td>Mon 2/11/13</td>
<td></td>
<td>Maria, Kenji</td>
</tr>
<tr>
<td>5</td>
<td>Prioritize requirements</td>
<td>2 days</td>
<td>Tue 2/12/13</td>
<td>Wed 2/13/13</td>
<td>2</td>
<td>Jason</td>
</tr>
<tr>
<td>6</td>
<td>Define Version 1 scope</td>
<td>2 days</td>
<td>Thu 2/14/13</td>
<td>Fri 2/15/13</td>
<td>5</td>
<td>Jason, Carly</td>
</tr>
<tr>
<td>7</td>
<td>Version 1</td>
<td>61 days</td>
<td>Mon 2/18/13</td>
<td>Mon 5/13/13</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>Detailed Requirements</td>
<td>17 days</td>
<td>Mon 2/18/13</td>
<td>Tue 3/12/13</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>Develop use cases</td>
<td>5 days</td>
<td>Mon 2/18/13</td>
<td>Fri 2/22/13</td>
<td>6</td>
<td>Ming</td>
</tr>
<tr>
<td>10</td>
<td>Develop process models</td>
<td>12 days</td>
<td>Mon 2/25/13</td>
<td>Tue 3/12/13</td>
<td>9</td>
<td>Ming</td>
</tr>
<tr>
<td>11</td>
<td>Develop data models</td>
<td>3 days</td>
<td>Mon 2/25/13</td>
<td>Wed 2/27/13</td>
<td>9</td>
<td>Maria</td>
</tr>
<tr>
<td>12</td>
<td>Preliminary Design</td>
<td>27 days</td>
<td>Mon 2/18/13</td>
<td>Tue 3/27/13</td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>System architecture</td>
<td>5 days</td>
<td>Mon 2/18/13</td>
<td>Fri 2/22/13</td>
<td>6</td>
<td>Kenji</td>
</tr>
<tr>
<td>14</td>
<td>User interface</td>
<td>7 days</td>
<td>Wed 3/17/13</td>
<td>Thu 3/21/13</td>
<td>10</td>
<td>Ming</td>
</tr>
<tr>
<td>15</td>
<td>Database</td>
<td>10 days</td>
<td>Thu 2/28/13</td>
<td>Wed 3/13/13</td>
<td>11</td>
<td>Maria</td>
</tr>
<tr>
<td>16</td>
<td>Programs</td>
<td>10 days</td>
<td>Wed 3/13/13</td>
<td>Tue 3/26/13</td>
<td>13, 10</td>
<td>Kenji</td>
</tr>
<tr>
<td>17</td>
<td>Implementation</td>
<td>56 days</td>
<td>Mon 2/25/13</td>
<td>Mon 5/13/13</td>
<td></td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>Acquire HW &amp; SW</td>
<td>10 days</td>
<td>Mon 2/25/13</td>
<td>Fri 3/8/13</td>
<td>13</td>
<td>Kenji</td>
</tr>
<tr>
<td>19</td>
<td>Construct database</td>
<td>7 days</td>
<td>Thu 3/14/13</td>
<td>Fri 3/22/13</td>
<td>15</td>
<td>Maria</td>
</tr>
<tr>
<td>20</td>
<td>Convert data</td>
<td>5 days</td>
<td>Mon 3/25/13</td>
<td>Fri 3/29/13</td>
<td>19</td>
<td>Maria</td>
</tr>
<tr>
<td>21</td>
<td>Write programs</td>
<td>20 days</td>
<td>Wed 3/27/13</td>
<td>Tue 4/23/13</td>
<td>16, 10</td>
<td>Ming, Kenji</td>
</tr>
<tr>
<td>22</td>
<td>Testing</td>
<td>10 days</td>
<td>Wed 4/24/13</td>
<td>Tue 5/7/13</td>
<td>20, 21</td>
<td>Maria, Kenji, Ming</td>
</tr>
<tr>
<td>23</td>
<td>Installation</td>
<td>4 days</td>
<td>Wed 5/8/13</td>
<td>Mon 5/13/13</td>
<td>22</td>
<td>Maria, Kenji, Ming</td>
</tr>
<tr>
<td>24</td>
<td>Version 2</td>
<td>42 days</td>
<td>Tue 5/14/13</td>
<td>Wed 7/12/13</td>
<td>23</td>
<td>Jason, Maria, Kenji, Ming</td>
</tr>
<tr>
<td>25</td>
<td>Version 3</td>
<td>28 days</td>
<td>Thu 7/11/13</td>
<td>Mon 8/19/13</td>
<td>24</td>
<td>Jason, Ming, Kenji, Maria</td>
</tr>
</tbody>
</table>

(Continued)
FIGURE 2-23
Gantt Chart
Chapter 2  Project Selection and Management

### FIGURE 2-24
Staffing Plan for the Digital Music Download System

<table>
<thead>
<tr>
<th>Role</th>
<th>Description</th>
<th>Assigned To</th>
</tr>
</thead>
<tbody>
<tr>
<td>Project manager</td>
<td>Oversees the project to ensure that it meets its objectives on time and within budget</td>
<td>Jason</td>
</tr>
<tr>
<td>Infrastructure analyst</td>
<td>Ensures that the system conforms to infrastructure standards at Tune Source; ensures that the Tune Source infrastructure can support the new system</td>
<td>Kenji</td>
</tr>
<tr>
<td>Systems analyst</td>
<td>Designs the information system—with a focus on interfaces with the CD sales system</td>
<td>Kenji</td>
</tr>
<tr>
<td>Systems analyst</td>
<td>Designs the information system—with a focus on the process models and interface design</td>
<td>Ming</td>
</tr>
<tr>
<td>Systems analyst</td>
<td>Designs the information system—with a focus on the data models and system performance</td>
<td>Maria</td>
</tr>
<tr>
<td>Programmer</td>
<td>Codes system</td>
<td>Ming</td>
</tr>
<tr>
<td>Programmer</td>
<td>Codes system</td>
<td>Kenji</td>
</tr>
</tbody>
</table>

Reporting structure: All project team members will report to Jason.

Special incentives: If the deadline for the project is met, all team members who contributed to this goal will receive a free day off, to be taken over the holiday season.

### FIGURE 2-25
Project Charter for the Digital Music Download System

**Project objective:** The Digital Music Download project team will create a working Web-based system to provide digital music downloads to Tune Source’s customers as rapidly as possible.

The Digital Music Download team members will

1. Attend a staff meeting each Friday at 2 P.M. to report on the status of assigned tasks.
2. Update the work plan with actual data each Friday by 5 P.M.
3. Discuss all problems with Jason as soon as they are detected.
4. Agree to support each other when help is needed, especially for tasks that could hold back the progress of the project.
5. Post important changes to the project on the team bulletin board as they are made.

### SUMMARY

**Project Selection**

Once the feasibility analysis has been completed, it is submitted back to the approval committee along with a revised system request. The committee then decides whether to approve the project, decline the project, or table it until additional information is available. The project selection process takes into account all of the projects in the organization, using portfolio management. The approval committee weighs many factors and makes trade-offs before a project is selected.
Creating the Project Plan

There are a number of different project methodologies that can be used to structure and guide systems development projects. Several of the key methodologies are waterfall development and its variations: parallel development and the V-model; rapid application development, including iterative development, system prototyping, and throwaway prototyping; and agile development, including extreme programming, Scrum, and others. The project manager evaluates characteristics of the project, including factors such as clarity of user requirements, familiarity with technology, complexity, reliability, time frame, and schedule visibility, to select the most appropriate methodology to use for the project.

The project manager then estimates the time frame for the project. Past experience, industry standards, and techniques such as function-point analysis, provide help in this task. The project methodology provides lists of tasks and deliverables for projects, which the project manager modifies, depending on the needs of the specific project. To create a work plan, the project manager refines the tasks into a work breakdown structure, and task time estimates and other information are entered into the work plan.

Staffing involves determining how many people should be assigned to the project, assigning project roles to team members, developing a reporting structure for the team, and matching people’s skills with the needs of the project. Staffing also includes motivating the team to meet the project’s objectives and minimizing conflict among team members. Both motivation and cohesiveness have been found to greatly influence performance of team members in project situations. Team members are motivated most by such nonmonetary things as recognition, achievement, and the work itself. Conflict can be minimized by clearly defining the roles on a project and holding team members accountable for their tasks. Some managers create a project charter that lists the project’s norms and ground rules.

Coordinating project activities includes putting efficient development practices in place and mitigating risk, and these activities occur over the course of the entire SDLC. Three techniques are available to help coordinate activities on a project: computer-aided software engineering (CASE), standards, and documentation. CASE is a category of software that automates all or part of the development process; standards are formal rules or guidelines that project teams must follow during the project; and documentation includes detailed information about the tasks of the SDLC. Often, documentation is stored in project binder(s) that contain all the deliverables and all the internal communication that takes place—the history of the project.

Managing and Controlling the Project

As the project progresses, the project manager collects status reports from the team members and updates the work plan. Graphical tools such as Gantt and PERT charts help depict progress on tasks and clarify critical task dependencies. Project managers try to avoid introducing scope creep or feature creep into the schedule. As inevitable project changes arise, however, project managers try to balance the project size (number of features), time frame, and cost. Estimates may have to be revised as more is learned about the system. Timeboxing can be used to deal with shortened time frames. The project manager also keeps a close watch on the project risk. A risk assessment should be created and updated to evaluate the likelihood of various risks and their potential impact on the project.
KEY TERMS

Agile development
CASE repository
Chief information officer
Computer-aided software engineering (CASE)
Design prototype
Documentation
Estimation
Feature creep
Functional lead
Group cohesiveness
Integrated CASE
Interpersonal skills
Iterative development
Lower CASE

Methodology
Milestones
Motivation
Parallel development
Project binder
Project management
Project manager
Project portfolio management
Rapid application development
Reporting structure
Risk assessment
Risk management
Scope creep
Staffing plan
Standards

System prototyping
Task dependencies
Technical skills
Technical lead
Throwaway prototyping
Timeboxing
Trade-offs
Upper CASE
V-model
Versions
Work breakdown structure
Work plan
Waterfall development

QUESTIONS

1. Describe how projects are selected in organizations.
2. Describe how project portfolio management is used by IT departments.
3. Describe the major elements and issues with waterfall development.
4. Describe the major elements and issues with parallel development.
5. Describe the major elements and issues with the V-model.
6. Describe the major elements and issues with iterative development.
7. Describe the major elements and issues with system prototyping.
8. Describe the major elements and issues with throwaway prototyping.
9. Describe the major elements and issues with agile development.
10. Compare and contrast structured design methodologies in general with rapid application development (RAD) methodologies in general.
11. Compare and contrast extreme programming and throwaway prototyping.
12. What are the key factors in selecting a methodology?
13. Why do many projects end up having unreasonable deadlines? How should a project manager react to unreasonable demands?
14. Name two ways to identify the tasks that need to be accomplished over the course of a project.
15. What is the difference between a methodology and a work plan? How are the two terms related?
16. Some companies hire consulting firms to develop the initial project plans and manage the project, but use their own analysts and programmers to develop the system. Why do you think some companies do this?
17. Describe the differences between a technical lead and a functional lead. How are they similar?
18. Describe three technical skills and three interpersonal skills that would be very important to have on any project.
19. What are the best ways to motivate a team? What are the worst ways?
20. List three techniques to reduce conflict.
21. What is the difference between upper CASE (computer-aided software engineering) and lower CASE?
22. Describe three types of standards, and provide examples of each.
23. What belongs in the project binder? How is the project binder organized?
24. What are the trade-offs that project managers must manage?
25. What is scope creep, and how can it be managed?
26. What is timeboxing, and why is it used?
27. Create a list of potential risks that could affect the outcome of a project.
28. Describe the factors that the project manager must evaluate when a project falls behind schedule.
APPENDIX 2A QUESTIONS

1. What is a function point, and how is it used?
2. Describe the three steps of the function point approach.
3. What is the formula for calculating the effort for a project?

APPENDIX 2B QUESTIONS

1. Compare and contrast the Gantt chart and the PERT chart.
2. Of what value is the Gantt chart to the project manager? The PERT chart?

EXERCISES

A. Suppose that you are a project manager using the waterfall development methodology on a large and complex project. Your manager has just read the latest article in Computerworld that advocates replacing the waterfall methodology with prototyping and comes to your office requesting you to switch. What do you say?
B. Suppose that you are an analyst developing a new information system to automate the sales transactions and manage inventory for each retail store in a large chain. The system would be installed at each store and would exchange data with a mainframe computer at the company’s head office. What methodology would you use? Why?
C. Suppose that you are an analyst developing a new executive information system (EIS) intended to provide key strategic information from existing corporate databases to senior executives to help in their decision making. What methodology would you use? Why?
D. Suppose that you are an analyst working for a small company to develop an accounting system. What methodology would you use? Why?
E. Visit a project management Web site, such as the Project Management Institute (www.pmi.org). Most have links to project management software products, white papers, and research. Examine some of the links for project management to better understand a variety of Internet sites that contain information related to this chapter.
F. Select a specific project management topic like computer-aided software engineering (CASE), project management software, or timeboxing, and use the Web search for information on that topic. The URL listed in question E or any search engine (e.g., Yahoo!, Google,) can provide a starting point for your efforts.
G. Pretend that the career services office at your university wants to develop a system that collects student résumés and makes them available to students and recruiters over the Web. Students should be able to input their résumé information into a standard résumé template. The information then is presented in a résumé format, and it also is placed in a database that can be queried through an online search form. You have been placed in charge of the project. Develop a plan for estimating the project. How long do you think it would take for you and three other students to complete the project? Provide support for the schedule that you propose.
H. Refer to the situation in question G. You have been told that recruiting season begins a month from today and that the new system must be used. How would you approach this situation? Describe what you can do as the project manager to make sure that your team does not burn out from unreasonable deadlines and commitments.
I. Consider the system described in question G. Create a work plan listing the tasks that will need to be completed to meet the project’s objectives. Create a Gantt chart and a PERT chart in a project management tool (e.g., Microsoft Project), or use a spreadsheet package to graphically show the high-level tasks of the project.
J. Suppose that you are in charge of the project described in question G, and the project will be staffed by members of your class. Do your classmates have all of the right skills to implement such a
project? If not, how will you go about making sure that the proper skills are available to get the job done?
K. Consider the application that is used at your school to register for classes. Complete a function point worksheet to determine the size of such an application. You will need to make some assumptions about the application’s interfaces and the various factors that affect its complexity.
L. Read “Your Turn 2-5” in Appendix 2A of this chapter. Create a risk assessment that lists the potential risks associated with performing the project, along with ways to address the risks.
M. Pretend that your instructor has asked you and two friends to create a Web page to describe the course to potential students and provide current class information (e.g., syllabus, assignments, readings) to current students. You have been assigned the role of leader, so you will need to coordinate your activities and those of your classmates until the project is completed. Describe how you would apply the project management techniques that you have learned in this chapter to this situation. Include descriptions of how you would create a work plan, staff the project, and coordinate all activities—yours and those of your classmates.

MINICASES

1. Emily Pemberton is an IS project manager facing a difficult situation. Emily works for the First Trust Bank, which has recently acquired the City National Bank. Prior to the acquisition, First Trust and City National were bitter rivals, fiercely competing for market share in the region. Following the acrimonious takeover, numerous staff were laid off in many banking areas, including IS. Key individuals were retained from both banks’ IS areas, however, and were assigned to a new consolidated IS department. Emily has been made project manager for the first significant IS project since the takeover, and she faces the task of integrating staffers from both banks on her team. The project they are undertaking will be highly visible within the organization, and the time frame for the project is somewhat demanding. Emily believes that the team can meet the project goals successfully, but success will require that the team become cohesive quickly and that potential conflicts are avoided. What strategies do you suggest that Emily implement in order to help ensure a successfully functioning project team?

2. Marcus Weber, IS project manager at ICAN Mutual Insurance Co., is reviewing the staffing arrangements for his next major project, the development of an expert system-based underwriters assistant. This new system will involve a whole new way for the underwriters to perform their tasks. The underwriters assistant system will function as sort of an underwriting supervisor, reviewing key elements of each application, checking for consistency in the underwriter’s decisions, and ensuring that no critical factors have been overlooked. The goal of the new system is to improve the quality of the underwriters’ decisions and to improve underwriter productivity. It is expected that the new system will substantially change the way the underwriting staff do their jobs.

Marcus is dismayed to learn that due to budget constraints, he must choose between one of two available staff members. Barry Filmore has had considerable experience and training in individual and organizational behavior. Barry has worked on several other projects in which the end users had to make significant adjustments
Appendix 2A—The Function Point Approach

The function point approach is an estimating technique that can be used to estimate the size of the new system, the effort that will be required to complete the system, and the time the project will require. This approach requires detailed knowledge of the system to be developed. When this knowledge is available, the function point approach produces a much more precise estimate for the project than the industry standard method mentioned earlier in Chapter 2.

The function point approach uses a three-step process (Figure 2A-1). First, the project manager estimates the size of the project in terms of the number of function points and lines of code. Then, the effort required (in person-months) is estimated. Finally, the time required (in months) is estimated.

Given that Marcus’s budget will permit him to add only Barry or Kim to this project team, but not both, what choice do you recommend for him? Justify your answer.

APPENDIX 2A—THE FUNCTION POINT APPROACH

lines of code the new system will require. This size estimate is then converted into the amount of effort required to develop the system in terms of the number of person-months. The estimated effort is then converted into an estimated schedule time in terms of the number of months from start to finish.

**Step 1: Estimate System Size** The first step is to estimate the size of a project by using function points, a concept developed in 1979 by Allen Albrecht of IBM. A function point is a measure of program size that is based on the system’s number and complexity of inputs, outputs, queries, files, and program interfaces.

To calculate the function points for a project, components are listed on a worksheet to represent the major elements of the system. For example, data-entry screens are kinds of inputs, reports are outputs, and database queries are kinds of queries. (See Figure 2A-2.) The project manager records the total number of each component that the system will include, and then he or she breaks down the number to show the number of components that have low, medium, and high complexity. In Figure 2A-2, there are 19 outputs that need to be developed for the system, 4 of which have low complexity, 10 that have medium complexity, and 5 that are very complex. After each line is filled in, a total number of points are calculated per line by multiplying each number by a complexity index. The complexity index values are drawn from function point research and tell us, for example, that a low complexity input is “worth” three function points, while a high complexity output is “worth” seven function points. The line totals are added up to determine the total unadjusted function points (TUFP) for the project.

The complexity of the overall system is greater than the sum of its parts. Things like the familiarity of the project team with the business area and the technology that will be used to implement the project also may influence how complex a project will be. A project that is very complex for a team with little experience might have little complexity for a team with lots of experience. To create a more realistic size for the project, a number of additional system factors such as end-user efficiency, reusability, and data communications are assessed in terms of their effect on the project’s complexity. (See Figure 2A-2.) These assessments are totaled and placed into a formula to calculate an adjusted project complexity (APC) factor. The APC factor has a baseline value of 0.65, and the total Processing Complexity (PC) score (converted to hundredths) is added to the baseline amount. The TUFP value is multiplied by the APC factor to determine the ultimate size of the project in terms of total adjusted function points (TAFP). This number should give the project manager a reasonable idea as to how big the project will be.

---

**Nielsen Media used function point analysis (FPA) for an upgrade to the Global Sample Management System (GSMS) for Nielsen Media/NetRatings, which keeps track of the Internet rating sample, a group of 40,000 homes nationwide that volunteer to participate in ongoing ratings.**

In late fall of 1998, Nielsen Media did an FP count based on the current GSMS. (FPA is always easier and more accurate when there is an existing system.) Nielsen Media had its counters—three quality assurance staff—do their FPA, and then input their count into Knowledge-Plan, a productivity modeling tool. In early 1999, seven programmers began writing code for the system, which they were expected to complete in 10 months. As November approached, the project was adding staff to try to meet the deadline. When it became evident that the deadline would not be met, a new FP count was conducted. The GSMS had grown to 900 FPs. Besides the original 500 plus 20%, there were 300 FPs attributable to features and functions that had crept into the project.

How did that happen? The way it always does: The developers and users had added a button here, a new feature there, and soon the project was much larger than it was originally. But Nielsen Media had put a stake in the ground at the beginning from which they could measure growth along the way.

The best practice is to run the FPA and productivity model at the project’s launch and again when there is a full list of functional requirements. Then do another analysis anytime there is a major modification in the functional definition of the project.

FIGURE 2A-2
Function Point-Estimation Worksheet

System Components:

<table>
<thead>
<tr>
<th>Description</th>
<th>Total Number</th>
<th>Complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Low</td>
</tr>
<tr>
<td>Inputs</td>
<td>6</td>
<td>3×3</td>
</tr>
<tr>
<td>Outputs</td>
<td>19</td>
<td>4×4</td>
</tr>
<tr>
<td>Queries</td>
<td>10</td>
<td>7×3</td>
</tr>
<tr>
<td>Files</td>
<td>15</td>
<td>0×7</td>
</tr>
<tr>
<td>Program Interfaces</td>
<td>3</td>
<td>1×5</td>
</tr>
<tr>
<td><strong>Total Unadjusted Function Points (TUFP):</strong></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Overall System:

<table>
<thead>
<tr>
<th>Feature</th>
<th>Complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data communications</td>
<td>3</td>
</tr>
<tr>
<td>Heavy use configuration</td>
<td>0</td>
</tr>
<tr>
<td>Transaction rate</td>
<td>0</td>
</tr>
<tr>
<td>End-user efficiency</td>
<td>0</td>
</tr>
<tr>
<td>Complex processing</td>
<td>0</td>
</tr>
<tr>
<td>Installation ease</td>
<td>0</td>
</tr>
<tr>
<td>Multiple sites</td>
<td>0</td>
</tr>
<tr>
<td>Performance</td>
<td>0</td>
</tr>
<tr>
<td>Distributed functions</td>
<td></td>
</tr>
<tr>
<td>Online data entry</td>
<td>2</td>
</tr>
<tr>
<td>Online update</td>
<td>0</td>
</tr>
<tr>
<td>Reusability</td>
<td>0</td>
</tr>
<tr>
<td>Operational ease</td>
<td>0</td>
</tr>
<tr>
<td>Extensibility</td>
<td>0</td>
</tr>
<tr>
<td><strong>Total Processing Complexity (PC):</strong></td>
<td></td>
</tr>
</tbody>
</table>

(0 = no effect on processing complexity; 3 = great effect on processing complexity)

**Adjusted Project Complexity (APC):**

\[
.65 + (0.01 \times 7) = .72
\]

**Total Adjusted Function Points (TAFP):**

\[
.72 \times 336 \text{(TUFP)} = 243 \text{(TAFP)}
\]
Sometimes a shortcut is used to determine the complexity of the project. Instead of calculating the exact APC score for the 14 factors listed in Figure 2A-2, project managers estimate an APC value that ranges from 0.65 for very simple systems to 1.00 for “normal” systems to as much as 1.35 for complex systems. This estimated APC score is then applied to the TUFSP to compute the TAFP. For example, a very simple system that has 200 unadjusted function points would have a size of 130 adjusted function points (200 / 0.65 = 130). However, if the system with 200 unadjusted function points were very complex, its function point size would be 270 (200 / 1.35 = 270).

In the planning phase, the exact nature of the system has not yet been determined, so it is impossible to know exactly how many inputs, outputs, and so forth will be in the system. It is up to the project manager to make an intelligent guess. Some people feel that using function points this early in a project is not practical for this reason. We believe function points can be a useful tool for understanding a project’s size at any point in the SDLC. Later in the project, once more is known about the system, the project manager will revise the estimates, using this better knowledge to produce more accurate results.

Once you have estimated the number of function points, you need to convert the number of function points into the lines of code that will be required to build the system. The number of lines of code depends on the programming language you choose to use. Figure 2A-3 presents a very rough conversion guide for some popular languages.

For example, the system in Figure 2A-2 has 243 function points. If you were to develop the system in COBOL, it would typically require approximately 26,730 lines of code to write it. Conversely, if you were to use Visual Basic, it typically would take 7290 lines of code. If you could develop the system by using a package such as Excel or Access, it would take between 2430 and 9720 lines of code. There is a great range for packages, because different packages enable you to do different things and not all systems can be built with certain packages. Sometimes you end up writing lots of extra code to do some simple function because the package does not have the capabilities you need.

There is also a very important message from the data in this figure. Since there is a direct relationship between lines of code and the amount of effort and time required to develop a system, the choice of development language has a significant impact on the time and cost of projects.
**Step 2: Estimate Effort Required** Once an understanding is reached about the size of the system, the next step is to estimate the effort that is required to build it. *Effort* is a function of the system size combined with production rates (how much work someone can complete in a given time). Much research has been done on software production rates. One of the most popular algorithms, the COCOMO model,\(^\text{21}\) was designed by Barry W. Boehm to convert a lines-of-code estimate into a person-month estimate. There are different versions of the COCOMO model that vary with the complexity of the software, the size of the system, the experience of the developers, and the type of software that you are developing (e.g., business application software such as the registration system at your university; commercial software such as Word; or system software such as Windows). For small to moderate-size business software projects (i.e., 100,000 lines of code and 10 or fewer programmers), the model is quite simple:

\[
\text{effort (in person-months)} = 1.4 \times \text{thousands of lines of code}
\]

For example, let’s suppose that we were going to develop a business software system requiring 10,000 lines of code. This project would typically take 14 person-months of effort. If the system in Figure 2A-2 were developed in COBOL (which equates to 26,730 lines of code), it would require about 37.42 person-months of effort.

![Table](image)

**FIGURE 2A-3**
Converting from Function Points to Lines of Code

**YOUR TURN 2A-2**

Refer to the project size and lines of code that you calculated in “Your Turn 2A-1.”

**QUESTIONS:**
1. Determine the effort of your project in person-months of effort by multiplying your lines of code (in thousands) by 1.4.
2. Calculate the schedule time in months for your project by using the formula \(3.0 \times \text{person-months}^{1/3}\).
3. Based on your numbers, how much time will it take to complete the project if you are the developer?

---

Step 3: Estimate Time Required Once the effort is understood, the optimal schedule for the project can be estimated. Historical data or estimation software can be used as aids, or one rule of thumb is to determine schedule by the following equation:

\[
\text{schedule time (months)} = 3.0 \times \text{person-months}^{1/3}
\]

This equation is widely used, although the specific numbers vary (e.g., some estimators may use 3.5 or 2.5 instead of 3.0). The equation suggests that a project that has an effort of 14 person-months should be scheduled to take a little more than 7 months to complete. Continuing the Figure 2A-2 example, the 37.42 person-months would require a little over 10 months. It is important to note that this estimate is for the analysis, design, and implementation phases; it does not include the planning phase.

APPENDIX 2B-PROJECT MANAGEMENT TOOLS: THE GANTT CHART AND PERT CHART

Project managers utilize several tools to help manage projects. The project work plan, discussed previously, is a critical element of managing projects. In addition, two graphical tools are widely used to understand the relationship between project tasks and to monitor progress on the project.

Gantt Chart

A Gantt chart is a horizontal bar chart that shows the same task information as the project work plan, but in a graphical way. Sometimes a picture really is worth a thousand words, and the Gantt chart can communicate the high-level status of a project much faster and easier than the work plan. Creating a Gantt chart is simple and can be done with a spreadsheet package, graphics software (e.g., Microsoft VISIO), or a project management package.

First, tasks are listed as rows in the chart, and time is listed across the top in increments based on the needs of the projects. (See Figure 2B-1). A short project may be divided into hours or days, whereas a medium-sized project may be represented in weeks or months. Horizontal bars are drawn to represent the duration of each task; the bar’s beginning and end mark exactly when the task will begin and end. As people work on tasks, the appropriate bars are filled in proportionately to how much of the task is finished. Too many tasks on a Gantt chart can become confusing, so it’s best to limit the number of tasks to around 20 to 30. If there are more tasks, break them down into subtasks and create Gantt charts for each level of detail.

There are many things a project manager can see by looking quickly at a Gantt chart. In addition to seeing how long tasks are and how far along they are, the project manager also can tell which tasks are sequential, which tasks occur at the same time, and which tasks overlap in some way. He or she can get a quick view of tasks that are ahead of schedule and behind schedule by drawing a vertical line on today’s date. If a bar is not filled in and appears to the left of the line, that task is behind schedule.

There are a few special notations that can be placed on a Gantt chart. Project milestones are shown by upside-down triangles or diamonds. Arrows are drawn between the task bars to show task dependencies. Sometimes, the names of people assigned to each task are listed next to the task bars to show what human resources have been allocated to each task.

PERT Chart

A second graphical way to look at the project work plan information is the PERT chart, which displays the project tasks in a flowchart. (See Figure 2B-2). PERT (Program Evaluation and Review Technique) is a network analysis technique that can be used when the individual task time estimates are fairly uncertain. Instead of assigning a specific value as the duration estimate, PERT uses three time estimates: optimistic, most likely, and pessimistic. It then combines the three estimates into a single weighted average estimate using the following formula:

\[
\text{PERT weighted average} = \frac{\text{optimistic value} + (4 \times \text{most likely value}) + \text{pessimistic value}}{6}
\]

The PERT chart is drawn graphically with boxes (called nodes) representing each task and lines (called arcs) showing the dependency between tasks. The time estimates are shown in the nodes. Usually, the partially completed tasks are displayed with a diagonal line through the node, and completed tasks contain crossed lines.
Appendix 2B—Project Management Tools: The Gantt Chart and Pert Chart

FIGURE 2B-1
Gantt Chart
**Design phase**
- Start: 11/15/13  ID: 1
- Finish: 12/27/13  Dur: 31 days
- Comp: 0%

**Develop database design document**
- Start: 12/2/13  ID: 2
- Finish: 12/12/13  Dur: 9 days
- Comp: 0%

**Develop rejects-handling design document**
- Start: 12/18/13  ID: 5
- Finish: 12/25/13  Dur: 9 days
- Comp: 0%

**Develop OLAP design document**
- Start: 12/13/13  ID: 7
- Finish: 12/25/13  Dur: 9 days
- Comp: 0%

**Develop OLAP design part 1**
- Start: 12/6/13  ID: 9
- Finish: 12/0/13  Dur: 8 days
- Comp: 0%

**Develop application design document**
- Start: 12/13/13  ID: 11
- Finish: 12/25/13  Dur: 9 days
- Comp: 0%

**ETL design document**
- Start: 12/26/13  ID: 13
- Finish: 12/27/13  Dur: 2 days
- Comp: 0%

**Application design document**
- Start: 11/15/13  ID: 15
- Finish: 12/19/13  Dur: 26 days
- Comp: 0%

**Functional requirements document**
- Start: 12/9/13  ID: 19
- Finish: 12/19/13  Dur: 9 days
- Comp: 0%
PERT charts are the best way to communicate task dependencies because they lay out the tasks in the order in which they need to be completed. The critical path method (CPM) allows the identification of the critical path in the network, the longest path from project inception to completion. The critical path shows all of the tasks that must be completed on schedule for the project as a whole to finish on schedule. If any of the tasks on the critical path (called critical tasks) takes longer than expected, the entire project will fall behind. CPM can be used with or without PERT.

Project management software packages like Microsoft Project enable the project manager to input the work plan once and then display the information in many different formats. You can toggle between the work plan, a Gantt chart, and a PERT chart, depending on your project management needs.