LEARNING OBJECTIVES

Once you have mastered the material in this chapter you will be able to:

1. Understand how projects are initiated and selected.
2. Define a business problem and determine the feasibility of a proposed project.
3. Plan a project by identifying activities and scheduling them.
4. Understand how an alternative approach called agile development balances objectives to manage the analysis and design process.
5. Manage team members and analysis and design activities so that the project objectives are met while the project remains on schedule.

Initiating projects, determining project feasibility, scheduling projects, and planning and then managing activities and team members for productivity are all important capabilities for the systems analyst to master. As such, they are considered project management fundamentals.

A systems project begins with problems or with opportunities for improvement in a business that often come up as the organization adapts to change. The increasing popularity of e-commerce means that some fundamental changes are occurring as businesses either originate their enterprises on, or move their internal operations as well as external relationships to, the Internet. Changes that require a systems solution occur in the legal environment as well as in the industry’s environment. Analysts work with users to create a problem definition reflecting current business systems and concerns. Once a project is suggested, the systems analyst works quickly with decision makers to determine whether it is feasible. If a project is approved for a full systems study, the project activities are scheduled through the use of tools such as Gantt charts and Program Evaluation and Review Techniques (PERT) diagrams so that the project can be completed on time. Part of assuring the productivity of systems analysis team members is effectively managing their scheduled activities. This chapter is devoted to a discussion of project management fundamentals.

PROJECT INITIATION

Systems projects are initiated by many different sources for many reasons. Some of the projects suggested will survive various stages of evaluation to be worked on by you (or you and your team); others will not and should not get that far.
Businesspeople suggest systems projects for two broad reasons: (1) because they experience problems that lend themselves to systems solutions, and (2) because they recognize opportunities for improvement through upgrading, altering, or installing new systems when they occur. Both situations can arise as the organization adapts to and copes with natural, evolutionary change.

PROBLEMS IN THE ORGANIZATION

Managers do not like to conceive of their organization as having problems, let alone talk about them or share them with someone from outside. Good managers, however, realize that recognizing symptoms of problems or, at a later stage, diagnosing the problems themselves and then confronting them are imperative if the business is to keep functioning at its highest potential.

Problems surface in many different ways. One way of conceptualizing what problems are and how they arise is to think of them as situations in which goals have never been met or are no longer being met. Useful feedback gives information about the gap between actual and intended performance. In this way feedback spotlights problems.

In some instances problems that require the services of systems analysts are uncovered because performance measures are not being met. Problems (or symptoms of problems) with processes that are visible in output and that could require the help of a systems analyst include excessive errors and work performed too slowly, incompletely, incorrectly, or not at all. Other symptoms of problems become evident when people do not meet baseline performance goals. Changes in employee behavior such as unusually high absenteeism, high job dissatisfaction, or high worker turnover should alert managers to potential problems. Any of these changes, alone or in combination, might be sufficient reason to request the help of a systems analyst.

Although difficulties such as those just described occur in the organization, feedback on how well the organization is meeting intended goals may come from outside, in the form of complaints or suggestions from customers, vendors, or suppliers, and lost or unexpectedly lower sales. This feedback from the external environment is extremely important and should not be ignored.

A summary of symptoms of problems and approaches useful in problem detection is provided in Figure 3.1. Notice that checking output, observing or researching employee behavior, and listening to feedback from external sources are all valuable in problem finding. When reacting to accounts of problems in the
Felix Straw, who represents one of the many U.S. distributors of the European soft drink Sipps, gazes unhappily at a newspaper weather map, which is saturated with dark red, indicating that most of the United States is experiencing an early spring heat wave with no signs of a letup. Pointing to the paper as he speaks, he tells your systems group, “It’s the best thing that could happen to us, or at least it should be. But when we had to place our orders three months ago, we had no idea that this spring monster heat wave was going to devour the country this way!” Nodding his head toward a picture of their European plant on the wall, he continues. “We need to be able to tell them when things are hot over here so we can get enough product. Otherwise, we’ll miss out every time. This happened two years ago and it just about killed us.

“Each of us distributors meets with our district managers to do three-month planning. When we agree, we fax our orders into European headquarters. They make their own adjustments, bottle the drinks, and then we get our modified orders about 9 to 15 weeks later. But we need ways to tell them what’s going on now. Why, we even have some new superstores that are opening up here. They should know we have extra-high demand.”

Corky, his assistant, agrees, saying, “Yeah, they should at least look at our past sales around this time of year. Some springs are hot, others are just average.”

Straw concurs, saying, “It would be music to my ears, it would be really sweet, if they would work with us to spot trends and changes—and then respond quickly.”

Stern’s, based in Blackpool, England, is a European beverage maker and the developer and producer of Sipps. Sipps is a sweet, fruit-flavored, nonalcoholic, noncarbonated drink, which is served chilled or with ice, and it is particularly popular when the weather is hot. Selling briskly in Europe and growing in popularity in the United States since its introduction five years ago, Sipps has had a difficult time adequately managing inventory and keeping up with U.S. customer demand, which is affected by seasonal temperature fluctuations. Places with year-round, warm-temperature climates and lots of tourists (such as Florida and California) have large standing orders, but other areas of the country could benefit from a less cumbersome, more responsive order-placing process. Sipps is distributed by a network of local distributors located throughout the United States and Canada.

As one of the systems analysts assigned to work with the U.S. distributors of Sipps, begin your analysis by listing some of the key symptoms and problems you have identified after studying the information flows, ordering process, and inventory management, and after interviewing Mr. Straw and his assistant. In a paragraph describe which problems might indicate the need for a systems solution.

Note: This consulting opportunity is loosely based on J. C. Perez, “Heineken’s HOPS Software Keeps A-Head on Inventory,” PC Week, Vol. 14, No. 2, January 13, 1997, pp. 31 and 34.
• Does the company plan to grow or expand?
• What is the business’s attitude (culture) about technology?
• What is the business’s budget for IT?
• Does the business’s staff have the expertise?

Needless to say, the systems analyst needs to understand how a business works.

The last part of the problem definition contains requirements, the things that must be accomplished, along with the possible solutions and the constraints that limit the development of the system. The requirements section may include security, usability, government requirements, and so on. Constraints often include the word “not,” indicating a limitation and may contain budget restrictions or time limitations.

The problem definition is produced after completing interviews, observations, and document analysis with the users. The result of gathering this information is a wealth of facts and important opinions in need of summary. The first step in producing the problem definition is to find a number of points that may be included in one issue. Major points can be identified in the interview in a number of ways:

1. Identifying an issue, topic, or theme that is repeated several times, sometimes by different people in several interviews.
2. Users may communicate the same metaphors, such as saying the business is a journey, war, game, organism, machine, and so on.
3. Users may speak at length on a topic.
4. Users may tell you outright that “This is a major problem.”
5. Users may communicate importance by body language or may speak emphatically on an issue.
6. The problem may be the first thing mentioned by the user.

Once the issues have been created, the objectives must be stated. At times the analyst may have to do a follow-up interview to obtain more precise information about the objectives. After the objectives are stated, the relative importance of the issues or objectives must be determined. If there are not enough funds to develop the complete system, the most critical objectives must be completed first. The identification of the most critical objectives is best done by users (with the support of analysts), because users are domain experts in their business area and in how they work best with technologies in the organization.

One technique is to ask the users to assign a weight for each issue or objective of the first draft of the problem definition. This is a subjective judgment by the user, but, if a number of users all assign weights and they are averaged together, the result might reflect the bigger picture. After the weights have been determined, the problem definition issues and objectives are resequenced in order of decreasing importance, the most important issues listed first. There is software such as Expert Choice (www.expertchoice.com) and other decision support software that can assist with the weighting and prioritizing of objectives.

Besides looking through data and interviewing people, try to witness the problem first hand. When looking at the same situation, an employee may view a problem very differently than a systems analyst does. This also gives analysts the opportunity to confirm their findings. In this way they use multiple methods, thereby strengthening the case for taking appropriate action.

**A Problem Definition Example: Catherine’s Catering**

Catherine’s Catering is a small business that caters meals, receptions, and banquets for business and social occasions such as luncheons and weddings. It was inspired by Catherine’s love of cooking and her talent for preparing fine meals. At first it was a small company with a handful of employees working on small projects. Catherine met with customers to
determine the number of people, the type of meals, and other information necessary to cater an event. As their reputation for creating superb food and the quality of the service began to blossom, the number of events started to increase. The building of a new convention center, along with a prospering business community in the city, increased the number of catering events.

Catherine was able to manage the business using spreadsheets and word processing but found difficulty in keeping up with endless phone calls about what types of meals were available, changes to the number of guests attending the event, and the availability of specialty dietary items, such as vegan, vegetarian, low-fat, low-carbohydrate, and so on. Catherine’s decisions to hire a number of part-time employees to cook and cater the events meant that the complexity of scheduling personnel was becoming overwhelming to the new human resources manager. Catherine decided to hire an IT and business consulting company to help her address the problems her catering enterprise was facing.

After performing interviews and observing a number of key staff, the consultants found the following concerns:

1. The master chef ordered supplies (produce, meat, and so on) from suppliers for each event. The suppliers would provide discounts if greater quantities were ordered at a single time for all events occurring in a given time frame.
2. Customers often called to change the number of guests for an event, with some changes made only one or two days before the event was scheduled.
3. It was too time-consuming for Catherine and her staff to handle each request for catering, with about 60 percent of the calls resulting in a contract.
4. Conflicts in employee schedules were occurring and some events were understaffed. Complaints about the timeliness of service were becoming more frequent.
5. Catherine does not have any summary information about the number of events and types of meals. It would be helpful to have trend information that would help guide her customers in their choice of meals.
6. Events are often held at hotels or other meeting halls, which provide table settings for sit-down meals. There are problems with having sufficient wait staff and changes with the number of guests.

The problem definition is shown in Figure 3.2. Notice the weights on the right, representing an average of the weights assigned by each employee. Objectives match the issues. Each objective is used to create user requirements.

User requirements are then used to create use cases and a use case diagram. Each objective may create one or more user requirement or several objectives may create one or perhaps no use cases (use cases are not often created for simple reports).

The user requirements for Catherine’s Catering are to:

1. Create a dynamic Web site to allow current and potential clients to view and obtain pricing information for a variety of different products.
2. Allow current and potential clients to submit a request with their catering choices, with the request routed to an account manager.
3. Add clients to the client database, assigning them a userID and a password for access to their projects.
4. Create a Web site for clients to view and update the number of guests for an event and restrict changing the number of guests when the event day is less than five days in the future.
5. Obtain or create software to communicate directly with event facility personnel.
Catherine's Catering

Problem Definition
Catherine's Catering is experiencing problems with handling the number of routine calls with customers, as well as coordinating with external partners such as suppliers and meeting facilities. The growth in the number of part-time staff is leading to scheduling conflicts and understaffed events.

Issues
1. Customer contact takes an inordinate amount of time for routine questions.
2. Managing part-time employees is time-consuming and leads to scheduling errors.
3. It is difficult to accommodate last-minute changes for events.
4. Supplies are ordered for each event. Often shipments are received several times a day.
5. There are often problems communicating changes to event facilities.
6. There is little historical information about customers and meals.

Objectives
1. Provide a Web system for customers to obtain pricing information and place orders.
2. Create or purchase a human resources system with a scheduling component.
3. After customers have signed an event contract, provide them with Web access to their account and a means for them to update the number of guests. Notify management of changes.
4. Provide a means to determine overall quantities of supplies for events occurring within a concurrent time frame.
5. Provide a system for communicating changes to key personnel at event facilities.
6. Store all event data and make summary information available in a variety of formats.

Requirements
1. The system must be secure.
2. Feedback must be entered by event managers at the close of each event.
3. There must be a means for event facilities to change their contact person.
4. The system must be easy to use by nontechnical people.

Constraints
1. Development costs must not exceed $50,000.
2. The initial Web site for customer orders must be ready by March 1 to accommodate requests for graduation parties and weddings.

6. Create or purchase a human resources system for scheduling part-time employees, allowing management to add employees and schedule them using a number of constraints.
7. Provide queries or reports with summary information.

SELECTION OF PROJECTS
Projects come from many different sources and for many reasons. Not all should be selected for further study. You must be clear in your own mind about the reasons for recommending a systems study on a project that seems to address a problem or could bring about improvement. Consider the motivation that prompts a proposal.
on the project. You need to be sure that the project under consideration is not being proposed simply to enhance your own political reputation or power, or that of the person or group proposing it, because there is a high probability that such a project will be ill-conceived and eventually ill-accepted.

As outlined in Chapter 2, prospective projects need to be examined from a systems perspective in such a way that you are considering the impact of the proposed change on the entire organization. Recall that the various subsystems of the organization are interrelated and interdependent, so a change to one subsystem might affect all the others. Even though the decision makers directly involved ultimately set the boundaries for the systems project, a systems project cannot be contemplated or selected in isolation from the rest of the organization.

Beyond these general considerations are five specific criteria for project selection:

1. Backing from management.
2. Appropriate timing of project commitment.
3. Possibility of improving attainment of organizational goals.
4. Practical in terms of resources for the systems analyst and organization.
5. Worthwhile project compared with other ways the organization could invest resources.

First and foremost is backing from management. Absolutely nothing can be accomplished without the endorsement of the people who eventually will foot the bill. This statement does not mean that you lack influence in directing the project or that people other than management can’t be included, but management backing is essential.

Another important criterion for project selection includes timing for you and the organization. Ask yourself and the others who are involved if the business is presently capable of making a time commitment for installation of new systems or improvement to existing ones. You must also be able to commit all or a portion of your time for the duration.

A third criterion is the possibility of improving attainment of organizational goals. The project should put the organization on target, not deter it from its ultimate goals.

A fourth criterion is selecting a project that is practicable in terms of your resources and capabilities as well as those of the business. Some projects will not fall within your realm of expertise, and you must be able to recognize them.

Finally, you need to come to a basic agreement with the organization about the worthiness of the systems project relative to any other possible project being considered. Remember that when a business commits to one project, it is committing resources that thereby become unavailable for other projects. It is useful to view all possible projects as competing for the business resources of time, money, and people.

**DETERMINING FEASIBILITY**

Once the number of projects has been narrowed according to the criteria discussed previously, it is still necessary to determine if the selected projects are feasible. Our definition of feasibility goes much deeper than common usage of the term, because systems projects feasibility is assessed in three principal ways: operationally, technically, and economically. The feasibility study is not a full-blown systems study. Rather, the feasibility study is used to gather broad data for the members of management that in turn enables them to make a decision on whether to proceed with a systems study.
Data for the feasibility study can be gathered through interviews, which are covered in detail in Chapter 4. The kind of interview required is directly related to the problem or opportunity being suggested. The systems analyst typically interviews those requesting help and those directly concerned with the decision-making process, typically management. Although it is important to address the correct problem, the systems analyst should not spend too much time doing feasibility studies, because many projects will be requested and only a few can or should be executed. The feasibility study must be highly time compressed, encompassing several activities in a short span of time.

**DEFINING OBJECTIVES**

The systems analyst serves as catalyst and supporting expert primarily by being able to see where processes can be improved. Optimistically, opportunities can be conceived of as the obverse of problems; yet, in some cultures, crisis also means opportunity. What looms as a disturbing problem for a manager might be turned into an opportunity for improvement by an alert systems analyst.

Improvements to systems can be defined as changes that will result in incremental but worthwhile benefits. There are many possibilities for improvements, including:

1. Speeding up a process.
2. Streamlining a process through the elimination of unnecessary or duplicated steps.
3. Combining processes.
4. Reducing errors in input through changes of forms and display screens.
5. Reducing redundant storage.
6. Reducing redundant output.
7. Improving integration of systems and subsystems.

It is well within the systems analyst’s capabilities to notice opportunities for improvements. People who come into daily contact with the system, however, may be even better sources of information about improvements that should be made. If improvements have already been suggested, your expertise is needed to help determine whether the improvement is worthwhile and how it can be implemented.

It is worthwhile for an analyst to create a feasibility impact grid (FIG) for understanding and assessing what impacts (if any) improvements to existing systems can make. Figure 3.3 maps out such a grid. The labels on the far left-hand side describe a variety of systems that exist currently or are proposed. They are categorized into three system types: ecommerce systems, management information systems (MIS), and transaction processing systems (TPS). Listed at the top are the seven process objectives. Red check marks in the grid show that a positive impact can be made when a system improvement is made. Green check marks indicate that the system has been implemented and that the improvement positively impacted the process objective.

Notice that the transaction processing systems show a positive effect on the process objectives in almost every case. Traditional management information systems may help make better decisions, but sometimes they do not help the efficient collecting, storing, or retrieving of data. Thus, there are fewer check marks in that part of the grid. When entering the world of ecommerce, the analyst needs to be aware of how each system improvement may effect process objectives. Notice that the analyst who completed this grid recognized that although some process objectives were effected, others were not.
Of equal importance is how corporate objectives are affected by improvements to information systems. These corporate objectives include:

1. Improving corporate profits.
2. Supporting the competitive strategy of the organization.
3. Improving cooperation with vendors and partners.
4. Improving internal operations support so that goods and services are produced efficiently and effectively.
5. Improving internal decision support so that decisions are more effective.
6. Improving customer service.
7. Increasing employee morale.

Once again a feasibility impact grid can be drawn to increase awareness of the impacts made on the achievement of corporate objectives. The grid shown in Figure 3.4 is similar to the process objectives grid described earlier, but it emphasizes the point that improvements to the management information system greatly affect corporate objectives. You may recall that traditional MIS did not affect many process objectives, but on the other hand, they do affect most of the corporate objectives.

It is essential that analysts systematically go through the steps in developing feasibility impact grids. By understanding process and corporate objectives, an analyst realizes why he or she is building systems and comprehends what the importance of
designing efficient and effective systems might be. Analysts can communicate those impacts to the decision makers evaluating (and paying for) the project.

An analyst should be aware that there also are some unacceptable objectives for systems projects. As mentioned before, they include undertaking a project solely to prove the prowess of the systems analysis team or purely to assert the superiority of one department over another in terms of its power to command internal resources. Without a consideration of its true contribution to the achievement of the organization’s goals, it is also unacceptable to automate manual procedures for the sake of automation alone or to invest in new technology because of infatuation with the "bells and whistles" it provides over and above what the present system offers.

The objectives of the project need to be cleared formally on paper as well as informally by talking to people in the business. Find out what problem they believe the systems project would solve or what situation it would improve, and what their expectations are for the proposed system.

DETERMINING RESOURCES

Resource determination for the feasibility study follows the same broad pattern discussed previously and will be revised and reevaluated if and when a formal systems study is commissioned. A project must be feasible in all three ways to merit designing efficient and effective systems.

An analyst can use a feasibility system component affects corporate objectives.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>▶️</td>
<td>Proposed information system component or improvement can contribute positively to the corporate objective.</td>
</tr>
<tr>
<td>▼️</td>
<td>Existing information system component is contributing positively to the corporate objective.</td>
</tr>
<tr>
<td>▼️</td>
<td>New information system component would not be feasible to implement in the future.</td>
</tr>
</tbody>
</table>

### FIGURE 3.4

An analyst can use a feasibility system component affects corporate objectives.
further development, as shown in Figure 3.5. Resources are discussed in relationship to three areas of feasibility: technical, economic, and operational.

**Technical Feasibility** A large part of determining resources has to do with assessing technical feasibility. The analyst must find out whether current technical resources can be upgraded or added to in a manner that fulfills the request under consideration. Sometimes, however, “add-ons” to existing systems are costly and not worthwhile, simply because they meet needs inefficiently. If existing systems cannot be added onto, the next question becomes whether there is technology in existence that meets the specifications.

At this point the expertise of systems analysts is beneficial, because by using their own experience and their contacts with vendors, systems analysts will be able to answer the question of technical feasibility. Usually the response to whether a particular technology is available and capable of meeting the users’ requests is “yes,” and then the question becomes an economic one.

**Economic Feasibility** Economic feasibility is the second part of resource determination. The basic resources to consider are your time and that of the systems analysis team, the cost of doing a full systems study (including the time of employees you will be working with), the cost of the business employee time, the estimated cost of hardware, and the estimated cost of software or software development.

The concerned business must be able to see the value of the investment it is pondering before committing to an entire systems study. If short-term costs are not overshadowed by long-term gains or produce no immediate reduction in operating costs, the system is not economically feasible and the project should not proceed any further.

**Operational Feasibility** Suppose for a moment that technical and economic resources are both judged adequate. The systems analyst must still consider the operational feasibility of the requested project. Operational feasibility is dependent on the human resources available for the project and involves projecting whether the system will operate and be used once it is installed.

If users are virtually wed to the present system, see no problems with it, and generally are not involved in requesting a new system, resistance to implementing the new system will be strong. Chances for it ever becoming operational are low.

Alternatively, if users themselves have expressed a need for a system that is operational more of the time, in a more efficient and accessible manner, chances are better that the requested system will eventually be used. Much of the art of
determining operational feasibility rests with the user interfaces that are chosen, as we see in Chapter 14.

At this point, determining operational feasibility requires creative imagination on the part of the systems analyst, as well as the powers of persuasion to let users know which interfaces are possible and which will satisfy their needs. The systems analyst must also listen carefully to what users really want and what it seems they will use. Ultimately, however, assessing operational feasibility largely involves educated guesswork.

**JUDGING FEASIBILITY**

From the foregoing discussion, it is evident that judging the feasibility of systems projects is never a clear-cut or easy task. Furthermore, project feasibility is a decision to be made not by the systems analyst but instead by management. Decisions are based on feasibility data expertly and professionally gathered and presented by the analyst.

The systems analyst needs to be sure that all three areas of technical, economic, and operational feasibility are addressed in the preliminary study. The study of a requested systems project must be accomplished quickly so that the resources devoted to it are minimal, the information output from the study is solid, and any existing interest in the project remains high. Remember that this is a preliminary study, which precedes the system study, and it must be executed rapidly and competently.

Projects that meet the criteria discussed in the “Selection of Projects” subsection earlier in the chapter, as well as the three criteria of technical, economic, and operational feasibility, should be chosen for a detailed systems study. At this point the systems analyst must act as a supporting expert, advising management that the requested systems project meets all the selection criteria and has thus qualified as an excellent candidate for further study. Remember that a commitment from management now means only that a systems study may proceed, not that a proposed system is accepted. Generally, the process of feasibility assessment is effective in screening out projects that are inconsistent with the business’s objectives, technically impossible, or economically without merit. Although it is painstaking, studying feasibility is worthwhile and saves businesses and systems analysts a good deal of time and money in the end.

**ACTIVITY PLANNING AND CONTROL**

Systems analysis and design involves many different types of activities that together make up a project. The systems analyst must manage the project carefully if the project is to be successful. Project management involves the general tasks of planning and control.

Planning includes all the activities required to select a systems analysis team, assign members of the team to appropriate projects, estimate the time required to complete each task, and schedule the project so that tasks are completed in a timely fashion. Control means using feedback to monitor the project, including comparing the plan for the project with its actual evolution. In addition, control means taking appropriate action to expedite or reschedule activities to finish on time while motivating team members to complete the job properly.

**ESTIMATING TIME REQUIRED**

The systems analyst’s first decision is to determine the amount of detail that goes into defining activities. The lowest level of detail is the systems development life
FOOD FOR THOUGHT

We could really make some changes. Shake up some people. Let them know we’re with it. Technologically, I mean,” said Malcolm Warner, vice president for AllFine Foods, a wholesale dairy products distributor. “That old system should be overhauled. I think we should just tell the staff that it’s time to change.”

“Yes, but what would we actually be improving?” Kim Han, assistant to the vice president, asks. “I mean, there aren’t any substantial problems with the system input or output that I can see.”

Malcolm snaps, “Kim, you’re purposely not seeing my point. People out there see us as a stodgy firm. A new computer system could help change that. Change the look of our invoices. Send jazzier reports to the food store owners. Get some people excited about us as leaders in wholesale food distributing and computers.”

“Well, from what I’ve seen over the years,” Kim replies evenly, “a new system is very disruptive, even when the business really needs it. People dislike change, and if the system is performing the way it should, maybe there are other things we could do to update our image that wouldn’t drive everyone nuts in the process. Besides, you’re talking big bucks for a new gimmick.”

Malcolm says, “I don’t think just tossing it around here between the two of us is going to solve anything. Check on it and get back to me. Wouldn’t it be wonderful?”

A week later Kim enters Malcolm’s office with several pages of interview notes in hand. “I’ve talked with most of the people who have extensive contact with the system. They’re happy, Malcolm. And they’re not just talking through their hats. They know what they’re doing.”

“I’m sure the managers would like to have a newer system than the guys at Quality Foods,” Malcolm replies. “Did you talk to them?”

Kim says, “Yes. They’re satisfied.”

“And how about the people in systems? Did they say the technology to update our system is out there?” Malcolm inquires insistently.

“Yes. It can be done. That doesn’t mean it should be,” Kim says firmly.

As the systems analyst for AllFine Foods, how would you assess the feasibility of the systems project Malcolm is proposing? Based on what Kim has said about the managers, users, and systems people, what seems to be the operational feasibility of the proposed project? What about the economic feasibility? What about the technological feasibility? Based on what Kim and Malcolm have discussed, would you recommend that a full-blown systems study be done? Discuss your answer in a paragraph.

cycle itself, whereas the highest extreme is to include every detailed step. The optimal answer to planning and scheduling lies somewhere in between.

A structured approach is useful here. In Figure 3.6 the systems analyst beginning a project has broken the process into three major phases: analysis, design, and implementation. Then the analysis phase is further broken down into data gathering, data flow and decision analysis, and proposal preparation. Design is broken down into data entry design, input and output design, and data organization. The implementation phase is divided into implementation and evaluation.

In subsequent steps the systems analyst needs to consider each of these tasks and break them down further so that planning and scheduling can take place. Figure 3.7 shows how the analysis phase is described in more detail. For example, data gathering is broken down into five activities, from conducting interviews to
observing reactions to the prototype. This particular project requires data flow analysis but not decision analysis, so the systems analyst has written in “analyze data flow” as the single step in the middle phase. Finally, proposal preparation is broken down into three steps: perform cost/benefit analysis, prepare proposal, and present proposal.

The systems analyst, of course, has the option to break down steps further. For instance, the analyst could specify each of the persons to be interviewed. The amount of detail necessary depends on the project, but all critical steps need to appear in the plans.

Sometimes the most difficult part of project planning is the crucial step of estimating the time it takes to complete each task or activity. When quizzed about reasons for lateness on a particular project, project team members cited poor scheduling estimates that hampered the success of projects from the outset. There is no substitute for experience in estimating time requirements, and systems analysts who have had the opportunity of an apprenticeship are fortunate in this regard.

Planners have attempted to reduce the inherent uncertainty in determining time estimates by projecting most likely, pessimistic, and optimistic estimates and then using a weighted average formula to determine the expected time an activity will take. This approach offers little more in the way of confidence, however. Perhaps the best strategy for the systems analyst is to adhere to a structured approach in identifying activities and describing these activities in sufficient detail. In this manner, the systems analyst will at least be able to limit unpleasant surprises.

**USING GANTT CHARTS FOR PROJECT SCHEDULING**

A Gantt chart is an easy way to schedule tasks. It is a chart on which bars represent each task or activity. The length of each bar represents the relative length of the task. Figure 3.8 is an example of a two-dimensional Gantt chart in which time is indicated on the horizontal dimension and a description of activities makes up the vertical dimension. In this example the Gantt chart shows the analysis or information gathering phase of the project. Notice on the Gantt chart that conducting interviews will take three weeks, administering the questionnaire will take four weeks, and so on. These activities overlap part of the time. In the chart the special
symbol ▲ signifies that it is week 9. The bars with color shading represent projects or parts of projects that have been completed, telling us that the systems analyst is behind in introducing prototypes but ahead in analyzing data flows. Action must be taken on introducing prototypes soon so that other activities or even the project itself will not be delayed as a result.

The main advantage of the Gantt chart is its simplicity. The systems analyst will find not only that this technique is easy to use but also that it lends itself to worthwhile communication with end users. Another advantage of using a Gantt chart is that the bars representing activities or tasks are drawn to scale; that is, the size of the bar indicates the relative length of time it will take to complete each task.

**USING PERT DIAGRAMS**

PERT is an acronym for Program Evaluation and Review Techniques. A program (a synonym for a project) is represented by a network of nodes and arrows that are then evaluated to determine the critical activities, improve the schedule if necessary, and review progress once the project is undertaken. PERT was developed in the late 1950s for use in the U.S. Navy’s Polaris nuclear submarine project. It reportedly saved the U.S. Navy two years’ development time.

PERT is useful when activities can be done in parallel rather than in sequence. The systems analyst can benefit from PERT by applying it to systems projects on a smaller scale, especially when some team members can be working on certain activities at the same time that fellow members are working on other tasks.

Figure 3.9 compares a simple Gantt chart with a PERT diagram. The activities expressed as bars in the Gantt chart are represented by arrows in the PERT diagram. The length of the arrows has no direct relationship with the activity durations. Circles on the PERT diagram are called events and can be identified by numbers, letters, or any other arbitrary form of designation. The circular nodes are present to (1) recognize that an activity is completed and (2) indicate which activities need to be completed before a new activity may be undertaken (precedence).

In reality activity C may not be started until activity A is completed. Precedence is not indicated at all in the Gantt chart, so it is not possible to tell whether activity C is scheduled to start on day 4 on purpose or by coincidence.

A project has a beginning, a middle, and an end; the beginning is event 10 and the end is event 50. To find the length of the project, each path from beginning to end is identified, and the length of each path is calculated. In this example path
PART I
SYSTEMS ANALYSIS FUNDAMENTALS

10–20–40–50 has a length of 15 days, whereas path 10–30–40–50 has a length of 11 days. Even though one person may be working on path 10–20–40–50 and another on path 10–30–40–50, the project is not a race. The project requires that both sets of activities (or paths) be completed; consequently, the project takes 15 days to complete.

The longest path is referred to as the critical path. Although the critical path is determined by calculating the longest path, it is defined as the path that will cause the whole project to fall behind if even one day’s delay is encountered on it. Note that if you are delayed one day on path 10–20–40–50, the entire project will take longer, but if you are delayed one day on path 10–30–40–50, the entire project will not suffer. The leeway to fall behind somewhat on noncritical paths is called slack time.

Occasionally, PERT diagrams need pseudo-activities, referred to as dummy activities, to preserve the logic of or clarify the diagram. Figure 3.10 shows two

FIGURE 3.9
Gantt charts compared with PERT diagrams for scheduling activities.

FIGURE 3.10
Precedence of activities is important in determining the length of the project when using a PERT diagram.
PERT diagrams with dummies. Project 1 and project 2 are quite different, and the way the dummy is drawn makes the difference clear. In project 1 activity C can only be started if both A and B are finished, because all arrows coming into a node must be completed before leaving the node. In project 2, however, activity C requires only activity B’s completion and can therefore be under way while activity A is still taking place.

Project 1 takes 14 days to complete, whereas project 2 takes only 9 days. The dummy in project 1 is necessary, of course, because it indicates a crucial precedence relationship. The dummy in project 2, on the other hand, is not required, and activity A could have been drawn from 10 to 40 and event 20 may be eliminated completely.

Therefore, there are many reasons for using a PERT diagram over a Gantt chart. The PERT diagram allows:
1. Easy identification of the order of precedence.
2. Easy identification of the critical path and thus critical activities.
3. Easy determination of slack time.

**A PERT Example** Suppose a systems analyst is trying to set up a realistic schedule for the data gathering and proposal phases of the systems analysis and design life cycle. The systems analyst looks over the situation and lists activities that need to be accomplished along the way. This list, which appears in Figure 3.11, also shows that some activities must precede other activities. The time estimates were determined as discussed in an earlier section of this chapter.

**Drawing the PERT Diagram** In constructing the PERT diagram, the analyst looks first at those activities requiring no predecessor activities, in this case A (conduct interviews) and C (read company reports). In the example in Figure 3.12, the analyst chose to number the nodes 10, 20, 30, and so on, and he or she drew two arrows out of the beginning node 10. These arrows represent activities A and C.
and are labeled as such. Nodes numbered 20 and 30 are drawn at the end of these respective arrows. The next step is to look for any activity requiring only A as a predecessor; task B (administer questionnaires) is the only one, so it can be represented by an arrow drawn from node 20 to node 30.

Because activities D (analyze data flow) and E (introduce prototype) require both activities B and C to be finished before they are started, arrows labeled D and E are drawn from node 30, the event that recognizes the completion of both B and C. This process is continued until the entire PERT diagram is completed. Notice that the entire project ends at an event called node 80.

Identifying the Critical Path Once the PERT diagram is drawn, it is possible to identify the critical path by calculating the sum of the activity times on each path and choosing the longest path. In this example, there are four paths: 10–20–30–50–60–70–80, 10–20–30–40–60–70–80, 10–30–50–60–70–80, and 10–30–40–60–70–80. The longest path is 10–20–30–50–60–70–80, which takes 22 days. It is essential that the systems analyst carefully monitor the activities on the critical path so as to keep the entire project on time or even shorten the project length if warranted.

COMPUTER-BASED PROJECT SCHEDULING

Using PCs for project scheduling has now become practical and straightforward. Microsoft Project is a good example of a powerful program.

An example of project management from Microsoft Project can be found in Figure 3.13. New tasks can be entered in either the top or the bottom part of the display, whichever is easier for the user. Let’s assume we want to enter the task “Conduct needs analysis” on the bottom half of the screen. First, we enter the name of the activity, then its duration, 5d (including a qualifier: d for day, w for week, etc.), and the ID for any predecessors. The ID, or identifier, is simply the
number of the task. We don’t have to enter a start date if we want the computer program to schedule it for us (as soon as possible, given the predecessors). The upper left part of the table lists the activities in the order in which we entered them. In the upper right a Gantt chart is shown.

Figure 3.14 is another display from Microsoft Project. The top half now shows a PERT diagram. Microsoft Project takes the liberty of representing the tasks or activities with rectangles rather than with arrows. Although this goes against the traditional conventions used in PERT diagrams, the software authors feel that it is easier to read tasks in rectangular boxes than to read them on arrows. A bold line, shown in red on the display, indicates the critical path. Once activities are drawn on the display, they can be repositioned using a mouse to enhance readability and communication with others. The dark box indicates that we are looking at that activity now. The dotted vertical line on the left side of the display shows the user where the page break will occur. The icons at the top of the page are familiar to anyone who uses Microsoft products.

TIMEBOXING

A recent development in project management is the concept of timeboxing. Traditionally, a project is broken down into phases, milestones, and tasks, but the timeboxing approach uses an absolute due date for the project and whatever has been accomplished by that due date is implemented. It is important to set a reasonable due date given the project size and goals. It is also important to prioritize the project goals so that the most important ones are delivered to the users by the due date. Lesser goals may be implemented later in the project. An example of timeboxing is creating a Web site that contains the most important features, with some of the minor pages containing an “Under Construction” image.

Other approaches to scheduling include integrated personal information managers, or PIMs. Some examples of a PIM include Microsoft Outlook and Palm Desktop. These PIMs are useful because they are a repository for phone and fax
numbers of business associates; for daily, weekly, or monthly planners; and for
to-do lists. Some PIMs are designed to be shells that enable you to launch other
programs and even allow you to store similar data from word processing and
spreadsheet programs in folders organized around a particular topic. Some are
good at sharing data with other programs, whereas others include Gantt charts to
aid in project management. Most PIMs can be synchronized with PIMs in Palm
computers and other handheld devices, cell phones, and watches, allowing for
excellent, wireless portability.

MANAGING THE PROJECT

The process of analysis and design can become unwieldy, especially when the sys-
tem being developed is large. To keep the development activities as manageable as
possible, we usually employ some of the techniques of project management to
help us get organized.

One important aspect of project management is how to manage one’s sched-
ule to finish the system on time, but it is not the only thing needed. The person in
charge, called the project manager, is often the lead systems analyst. The project
manager needs to understand how to determine what is needed and how to initi-
ate a project; how to develop a problem definition; how to examine feasibility of
completing the systems project; how to reduce risk; how to identify and manage
activities; and how to hire, manage, and motivate other team members.

ADDRESSING SYSTEM COMPLEXITY

Estimating models, such as Costar (www.softstarsystems.com) or Construx
(http://www.construx.com) work as follows: First the systems analyst enters an
estimate of the size of the system. This can be entered in a number of different
ways, including the lines of source code of the current system. Then it may be
helpful to adjust the degree of difficulty based on how familiar the analyst is with
this type of project.

Also considered are other variables, like the experience or capability of the
team, the type of platform or operating system, the level of usability of the fin-
ished software (for example, what languages are necessary), and other factors that
can drive up costs. Once the data are entered, calculations are made, and a rough
projection of the completion date is produced. As the project gets underway, more
specific estimates are possible.

FUNCTION POINT ANALYSIS

You can see that experience has a lot to do with shaping the estimates. If a project
manager keeps track of the time it takes each member of the team to complete
specific activities, then in future projects it becomes much easier to come up with
accurate estimates. As mentioned previously, one way to predict the time it will
take to complete a systems project is to estimate the necessary lines of code
directly. Another way to estimate size is by using function point analysis. Function
point analysis is based on five main components of computers systems.

1. External inputs.
2. External outputs.
3. External queries.
4. Internal logical files.
5. External interface files.
To determine the function point count (FPC), count the preceding components, rate each of them in terms of complexity, assign numbers to them based on their complexity, arrive at a subtotal, then multiply that number by an adjustment factor based on systems characteristics that make the system more complex. The entire process is accomplished using rules of thumb, so the function point count becomes an estimator of how complex a project will be. In this book, we use estimates or rules of thumb developed by the International Function Point User Group. To understand more about function points, go to www.ifpug.org.

As you can see immediately, it is first necessary to define a boundary (as covered in Chapter 2) so that we know what is internal and what is external to the system. Once this is accomplished, the components can be placed into one of the preceding categories.

An external input can be data from another application or even data entered on an input screen. External outputs are usually derived variables, calculated within the system, that transfer to another system or are used to prepare a report. External queries are actions that retrieve data from internal files or databases but do not change them.

Internal logical files are clusters of logically identifiable data that are stored entirely within the system. Finally, external interface files are groups of logically related data maintained by applications outside the system and used for reference purposes only. (Note that in these last two instances we are referring to logical, not physical, groups and storage. Where data are physically stored does not matter.)

For example, a spell checker would have external input keyed in by a user. External output would include an error message with suggested alternative spellings. An external query is a lookup table for each word against the internal dictionary. The standard dictionary and a second, user-entered dictionary would be included as internal logical files. An example of an external interface file would be an external dictionary, such as www.wiktionary.org, that could be referenced if the user required more information to answer a question such as, “Is JAVA (all caps) or Java (upper and lower case) the correct way to express the programming language?” (The correct answer is “Java.”)

Calculating function point values is a five-step process, as shown in Figure 3.15.

1. Determine the complexity (low, average, or high) of external inputs, external outputs, and external queries based on the number of files updated or referenced and the number of different data element types. (For now picture “data elements” as the last name, first name, address, city, etc.) Assume that two or three files updated or referenced and 6 to 20 data elements are “average.” If the number of either files or data elements increases, the complexity increases to “high.” If the number decreases, the complexity decreases to “low.”

Therefore, if an external input referenced two files and there are 10 data elements, a Step 1 table lookup would determine that this external input was rated “average.” If an external output referenced two files and there are 25 data elements, a Step 1 table lookup would determine that this external input was rated “high.” This table lookup would be repeated for each external input, external output, and external query.

2. Determine the complexity (low, average, or high) of internal logical files and external interface files based on the number of groups of data elements and the number of different element types. (For example, a person resides at an address that is composed of a group of data elements including last name, first name, address, city, state, and so on.) Assume two to five logical groups of data elements and 20 to 50 different data element types are average. If either the
**FIGURE 3.15**
Function point counts can be accomplished in five steps.

**STEP 1:** Look up low, average, and high values for external inputs, external outputs, and external queries.

<table>
<thead>
<tr>
<th>File Types</th>
<th>Number of Data Elements Included</th>
</tr>
</thead>
<tbody>
<tr>
<td>Referenced</td>
<td>1–5, 6–20, 21+</td>
</tr>
<tr>
<td>0</td>
<td>Low</td>
</tr>
<tr>
<td>1–3</td>
<td>Average</td>
</tr>
<tr>
<td>4+</td>
<td>High</td>
</tr>
</tbody>
</table>

**STEP 2:** Look up low, average and high values for internal logical files and external interface files.

<table>
<thead>
<tr>
<th>Group of Elements</th>
<th>Number of Data Elements Included</th>
</tr>
</thead>
<tbody>
<tr>
<td>0–1</td>
<td>1–19, 20–50, 51+</td>
</tr>
<tr>
<td>2–5</td>
<td>Low, Average, High</td>
</tr>
<tr>
<td>6+</td>
<td>Average, High, High</td>
</tr>
</tbody>
</table>

**STEP 3:** Calculate the number of unadjusted functions points (UFP) using predetermined weights.

<table>
<thead>
<tr>
<th>Type of Component</th>
<th>Number of Components</th>
<th>Complexity of Components</th>
<th>Components</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>External inputs</td>
<td>5, 2, 1, 2</td>
<td>2 × 3</td>
<td>1 × 4</td>
<td>2 × 6</td>
</tr>
<tr>
<td>External outputs</td>
<td>12, 4, 6, 2</td>
<td>4 × 4</td>
<td>6 × 5</td>
<td>2 × 7</td>
</tr>
<tr>
<td>External queries</td>
<td>20, 5, 10, 5</td>
<td>5 × 3</td>
<td>10 × 4</td>
<td>5 × 6</td>
</tr>
<tr>
<td>Internal logical files</td>
<td>13, 3, 5</td>
<td>3 × 7</td>
<td>5 × 10</td>
<td>5 × 15</td>
</tr>
<tr>
<td>External interface files</td>
<td>2, 1, 0, 1</td>
<td>1 × 5</td>
<td>0 × 7</td>
<td>1 × 10</td>
</tr>
</tbody>
</table>

Total unadjusted function points: 328

**STEP 4:** Determine the value adjustment factor (VAF) by rating each system characteristic and calculating a subtotal, then dividing it by 100.

<table>
<thead>
<tr>
<th>System Characteristic</th>
<th>Rating</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data communications</td>
<td>2</td>
</tr>
<tr>
<td>Distributed data processing</td>
<td>2</td>
</tr>
<tr>
<td>Performance</td>
<td>0</td>
</tr>
<tr>
<td>Heavily used configuration</td>
<td>0</td>
</tr>
<tr>
<td>Transaction rate</td>
<td>0</td>
</tr>
<tr>
<td>Online data entry</td>
<td>0</td>
</tr>
<tr>
<td>End user efficiency</td>
<td>0</td>
</tr>
<tr>
<td>Online update</td>
<td>0</td>
</tr>
<tr>
<td>Complex processing</td>
<td>0</td>
</tr>
<tr>
<td>Reusability</td>
<td>2</td>
</tr>
<tr>
<td>Installation ease</td>
<td>3</td>
</tr>
<tr>
<td>Operational ease</td>
<td>3</td>
</tr>
<tr>
<td>Multiple sites</td>
<td>3</td>
</tr>
<tr>
<td>Facilitate change</td>
<td>1</td>
</tr>
</tbody>
</table>

VAF = Total divided by 100 = 16/100 = 0.16

**STEP 5:** Calculate the number of adjusted function points using the following formula:

\[ FPC = UFP \times (0.65 + VAF) = 328 \times (0.65 + 0.16) = 328 \times 0.81 = 266 \]
groups of data elements or the data elements increase, the complexity increases to "high." If either decrease, the complexity decreases to "low."

Therefore, if an external logical file contains only one group of data elements and there are 25 different types of data elements, a Step 2 table lookup would determine that this external logical file was rated "low." This table lookup would be repeated for each internal logical file and external interface file.

3. Calculate the number of unadjusted function points (UFP) by assigning predetermined weights (numbers) to low, medium, and high complexity for each of the five components and calculate the result. For example, the weights for external inputs were assigned values of low = 3, average = 4, and high = 6. Then multiple the number of external inputs (2 of low complexity, 1 of average, and 2 of high) times these weights. Subtotal them. Once you have done this for all of the components, sum the subtotals to calculate the number of unadjusted function points.

4. Determine the value adjustment factor (VAF) based on 14 general characteristics of the system. To complete this table, you may need to ask questions: "How are distributed data functions handled?" "Does the application have extensive logic processing?" "To what extent are start-up, backup, and recovery procedures automatic?" These questions allow you to assign a rating of 0 (no effect on the complexity) to 3 (great effect on the complexity). Once you do this for all of the standard 14 characteristics, calculate the total and divide by 100 to get the value adjustment factor (VAF).

5. Calculate the function point count (FPC) by multiplying the number of unadjusted function points (UAF in Step 3) by the total of 0.65, plus the value adjustment factor (VAF in Step 4).

   Note that 0.65 (or 65 percent) has been predetermined and represents the multiplier when there is no effect on complexity. The VAF is a number that can range between 0.00 and 0.70 (if all 14 systems characteristics are rated as 5, having a strong effect on complexity). Therefore the number you multiply by the UAF is always a number between 0.65 (if system characteristics have no effect on complexity) and 1.35 (if system characteristics have a strong effect on complexity). If all of the system characteristics have an average effect on complexity, the VAF would be 1.02.

   The function point count can be used to estimate lines of code and differs for various languages. Software Product Research (www.spr.com/products/programming.shtm) claims that, if a developer is using a second-generation language, the default is to multiply the number of function points by 107, a third-generation language by 80, and a fourth generation language by 20. In our example we determined that the function point count was 266, so using COBOL our system might require approximately 28,000 lines of code, Java or C++ may require 21,000 lines, and PowerBuilder or HTML might require only 5,000 lines.

   This approach has merit, but only if the project manager has prior experience developing systems. Otherwise, it is very difficult to estimate how many external inputs, outputs, and the like are created in developing the system. After the system development is underway, it becomes a bit easier to estimate the number of these components and consequently this approach has more value.

ESTIMATING STAFFING REQUIREMENTS FOR THE PROJECT

One of the key lessons from the last section is to make sure you are alert to how much your choice of software can influence the amount of effort that goes into
system development. If COBOL is chosen, the team will be responsible for writing 28,000 lines of code, but if PowerBuilder is chosen, the project could be done in about 5,000 lines. The choice of language may have an enormous effect on the time it takes to complete the project and the number of analysts and programmers needed to complete the work.

A rule of thumb commonly used to estimate a project in person-months is:

\[
\text{Number of person months} = 1.4 \times \frac{\text{Number of lines of code}}{1,000}
\]

Therefore, our system would take \((1.4 \times 28,000/1,000) = 39.2\) or about 39 months to develop the system in COBOL, 29 months to finish it in C++ or Java, and only seven months to complete the project in PowerBuilder.

But, once again, project managers who are experienced in thinking in terms of person-months realize that it does not work to say, “If it takes seven months to finish a project, let’s hire seven people and complete it in one month!” Picture the following. If you wanted to replace a roof, one person can do the work, but two can do it faster. Three may be faster yet. But teams of people need to communicate. Otherwise, they get in each other’s way. Maybe one of them even falls off the roof. This principle applies to systems projects as well. When two systems developers are involved, there is one set of two-way communication. If three are involved there are three sets. If four developers are on the project, there are 12 sets of two-way communication, and so on.

One rule of thumb that used for calculating months scheduled is:

\[
\text{Scheduled months} = 3 \times \text{Person months}^{1/3}
\]

For our example of a project using COBOL:

\[
\text{Scheduled months} = 3 \times 39^{1/3} = 3 \times 3.39 = 10.17 = \text{or about 10 months}
\]

Using C++ or Java:

\[
\text{Scheduled months} = 3 \times 29^{1/3} = 3 \times 3.07 = 9.21 = \text{approximately 9 months}
\]

Using PowerBuilder:

\[
\text{Scheduled months} = 3 \times 7^{1/3} = 3 \times 1.91 = 5.73 = \text{approximately 6 months}
\]

(Note that the function in Microsoft Excel for the cube root or raising to the 1/3 power is \([\text{CellName}]^3\).)

So the project manager would need to allow from 6 to 10 months, depending on the language selected to perform the analysis, design, and implementation phases of the project. The project manager would need to add additional time for planning at the beginning and for training and maintenance at the end.

**Managing Risk**

Although much effort is put into function point analysis to arrive at reasonable estimates, many projects actually fail. Studies have shown that about 30 percent of
all projects succeed (that is, they were delivered with all the necessary features, on time, and within budget), but as many as 20 percent fail (were never completed or used). The remaining 50 percent of the projects are either late, over budget, or offer fewer features than originally promised.

The system we looked at earlier in this chapter had a function point count of only 266. In that range, projects terminate prior to completion at a rate of about 10 percent. Large systems projects (those with 10,000 lines of code) are less likely to be completed, with a rate of failure of about 50 percent. Project managers must realize they need to break complex projects down into smaller projects to increase their probability of success.

Projects often fail because things happen unexpectedly, but a good project manager can identify potential problems. For example, if some members of the team are currently wrapping up another project, a project manager can assume some probability that the teams members may take longer than expected and not be able to give full time to the new project. Suppose the probability of this is 50 percent and, if it happens, it would delay the project 30 days (6 work weeks).

Another possibility is that key managers would not be available to interview during the needs requirement phase because of other work on their agenda. Suppose this had a probability of 25 percent and, if it occurs, it would cause a delay of 20 days.

By carefully thinking about scenarios that could cause potential problems and calculating the expected value of all delays, the project manager is able to add additional time as a buffer to protect against the entire project failing. Figure 3.16 shows how to list all the potential problems and calculate the extra time that should be budgeted if the project manager wants to ensure the project is completed on time.

### MANAGING ANALYSIS AND DESIGN ACTIVITIES

Along with managing time and resources, systems analysts must also manage people. Management is accomplished primarily by communicating accurately to team members who have been selected for their competency and compatibility. Goals for project productivity must be set, and members of systems analysis teams must be motivated to achieve them.

### ASSEMBLING A TEAM

Assembling a team is desirable. If a project manager has the opportunity to create a dream team of skilled people to develop a system, whom should they choose? In general, project managers need to look for others who share their values of teamwork guided by the desire to deliver a high-quality system on time and on budget. Other desirable team member characteristics include a good work ethic, honesty,
competency; a readiness to take on leadership based on expertise; motivation, enthusiasm for the project, and trust of teammates.

The project manager needs to know about business principles, but it doesn’t hurt to have at least one other person on the team who understands how a business operates. Perhaps this person should be a specialist in the same area as the system being developed. When developing an ecommerce site, teams can enlist the help of someone in marketing; those developing an inventory system can ask a person versed in production and operations to provide expertise.

A team ideally should have two systems analysts on it. They can help each other, check each other’s work, and shift their workloads accordingly. There is certainly a need to have people with programming skills on board. Coding is important, but people who know how to conduct walk-throughs, reviews, testing, and documenting systems are important as well. Some people are good at seeing the big picture, while others perform well when tasks are broken down into smaller ones for them. Every team should have both types of individuals.

Beyond the basics, a project manager should look for people with both experience and enthusiasm. Experience is especially important when trying to estimate the time required to complete a project. Experience in programming can mean code is developed five times faster than if it is developed by an inexperienced team. A usability expert is also a useful addition to the team.

The team must be motivated. One way to keep the team positively oriented throughout the entire process is to select good people at the outset. Look for enthusiasm, imagination, and an ability to communicate with different kinds of people. These basic attributes hold the potential for success. It also helps to hire superior writers and articulate speakers who can present proposals and work directly with customers.

Trust is an important part of a team. All members of the project need to act responsibly and agree to do their best and complete their part of the project. People may have different work styles, but they all need to agree to work together toward a common goal.

COMMUNICATION STRATEGIES FOR MANAGING TEAMS

Teams have their own personalities, a result of combining each individual team member with every other in a way that creates a totally new network of interactions. A way to organize your thinking about teams is to visualize them as always seeking a balance between accomplishing the work at hand and maintaining the relationships among team members.

In fact, teams will often have two leaders, not just one. Usually one person will emerge who leads members to accomplish tasks, and another person will emerge who is concerned with the social relationships among group members. Both are necessary for the team. These individuals have been labeled by other researchers as, respectively, task leader and socioemotional leader. Every team is subject to tensions that are an outgrowth of seeking a balance between accomplishing tasks and maintaining relationships among team members.

For the team to continue its effectiveness, tensions must be continually resolved. Minimizing or ignoring tensions will lead to ineffectiveness and eventual disintegration of the team. Much of the tension release necessary can be gained through skillful use of feedback by all team members. All members, however, need to agree that the way they interact (i.e., process) is important enough to merit some time. Productivity goals for processes are discussed in a later section.

Securing agreement on appropriate member interaction involves creating explicit and implicit team norms (collective expectations, values, and ways of
behaving) that guide members in their relationships. A team’s norms belong to it and will not necessarily transfer from one team to another. These norms change over time and are better thought of as a team process of interaction rather than a product.

Norms can be functional or dysfunctional. Just because a particular behavior is a norm for a team does not mean it is helping the team to achieve its goals. For example, an expectation that junior team members should do all project scheduling may be a team norm. By adhering to this norm, the team is putting extreme pressure on new members and not taking full advantage of the experience of the team. It is a norm that, if continued, could make team members waste precious resources.

Team members need to make norms explicit and periodically assess whether norms are functional or dysfunctional in helping the team achieve its goals. The overriding expectation for your team must be that change is the norm. Ask yourself whether team norms are helping or hindering the team’s progress.

SETTING PROJECT PRODUCTIVITY GOALS

When you have worked with your team members on various kinds of projects, you or your team leader will acquire acumen for projecting what the team can achieve in a specific amount of time. Using the hints discussed in the earlier section in this chapter on methods for estimating time required and coupling them with experience will enable the team to set worthwhile productivity goals.

Systems analysts are accustomed to thinking about productivity goals for employees who show tangible outputs, such as the number of blue jeans sewn per hour, the number of entries keyed in per minute, or the number of items scanned per second. As manufacturing productivity rises, however, it is becoming clear that managerial productivity must keep pace. It is with this aim in mind that productivity goals for the systems analysis team are set.
Goals need to be formulated and agreed to by the team, and they should be based on team members’ expertise, former performance, and the nature of the specific project. Goals will vary somewhat for each project undertaken, because sometimes an entire system will be installed, whereas other projects might involve limited modifications to a portion of an existing system.

**MOTIVATING PROJECT TEAM MEMBERS**

Although motivation is an extremely complex topic, it is a good one to consider, even if briefly, at this point. To oversimplify, recall that people join organizations to provide for some of their basic needs such as food, clothing, and shelter. All humans, however, also have higher-level needs, which include affiliation, control, independence, and creativity. People are motivated to fulfill unmet needs on several levels.

Team members can be motivated, at least partially, through participation in goal setting, as described in the previous section. The very act of setting a challenging but achievable goal and then periodically measuring performance against the goal seems to work in motivating people. Goals act almost as magnets in attracting people to achievement.

Part of the reason goal setting motivates people is that team members know prior to any performance review exactly what is expected of them. The success of goal setting for motivating can also be ascribed to it, affording each team member some autonomy in achieving the goals. Although a goal is predetermined, the means to achieve it may not be. In this instance team members are free to use their own expertise and experience to meet their goals.

Setting goals can also motivate team members by clarifying for them and others what must be done to get results. Team members are also motivated by goals because goals define the level of achievement that is expected of them. This use of goals simplifies the working atmosphere, but it also electrifies it with the possibility that what is expected can indeed be done.

**MANAGING PROJECTS USING COTS SOFTWARE**

Sometimes commercial off-the-shelf (COTS) software is used to finish the project faster or to decrease the risk involved. Managing such projects still requires careful planning.

Some people define COTS software very broadly. That is, they consider a wide range of packages such as Microsoft Word and Microsoft Access to be COTS software packages. COTS software for the PC would therefore include off-the-shelf virus protection, graphical software, and income tax packages. Others define it as industry-specific software. The result is the same—rather than writing your own code, you can simply adopt these packages.

COTS software packages allow some customization. Macros and templates can be used to customize them to a particular business. COTS software packages, however, often pose compatibility problems and do not work well together. Before Windows XP, in fact, the installation of some packages disabled others (the authors have suffered through this problem themselves many times). But even now, two packages offered by WordPerfect Corporation (CorelDraw Graphics Suite and Corel Designer) have keystrokes and commands that are not shared, but rather independent of one another. Because one of the arguable advantages of COTS software packages includes the ability to train people easily, this lack of common keystrokes and commands is a contradiction. Other COTS software packages for decision support will be discussed in Chapter 10.
MANAGING ECOMMERCE PROJECTS

Many of the approaches and techniques discussed earlier are transferable to e-commerce project management. You should be cautioned, however, that although there are many similarities, there are also many differences. One difference is that the data used by ecommerce systems are scattered all over the organization. Therefore, you are not just managing data in a self-contained department or even one solitary unit. Hence, many organizational politics can come into play, because units often feel protective of the data they generate and do not understand the need to share them across the organization.

Another stark difference is that ecommerce project teams typically need more staff with a variety of skills, including developers, consultants, database experts, and system integrators, from across the organization. Neatly defined, stable project groups that exist within a cohesive IS group or systems development team will be the exception rather than the rule. In addition, because so much help may be required initially, ecommerce project managers need to build partnerships externally and internally well ahead of the implementation, perhaps sharing talent across projects to defray costs of ecommerce implementations and to muster the required numbers of people with the necessary expertise. The potential for organizational politics to drive a wedge between team members is very real.

One way to prevent politics from sabotaging a project is for the ecommerce project manager to emphasize the integration of the ecommerce with the organization’s internal systems and in so doing emphasize the organizational aspect embedded in the ecommerce project. As one ecommerce project manager told us, “Designing the front end [what the consumer sees] is the easy part of all this. The real challenge comes from integrating ecommerce strategically into all the organization’s systems.”

A fourth difference between traditional project management and ecommerce project management is that because the system will be linking with the outside world via the Internet, security is of the utmost importance. Developing and implementing a security plan before the new system is in place is a project in and of itself and must be managed as such.

CREATING THE PROJECT CHARTER

Part of the planning process is to agree on what will be done and at what time. Analysts who are external consultants, as well as those who are organization members, need to specify what they will eventually deliver and when they will deliver it. This chapter has elaborated on ways to estimate the delivery date for the completed system and also how to identify organizational goals and assess the feasibility of the proposed system.

The project charter is a written narrative that clarifies the following questions:

1. What does the user expect of the project (what are the objectives)? What will the system do to meet the needs (achieve the objectives)?
2. What is the scope (or what are the boundaries) of the project? (What does the user consider to be beyond the project’s reach?)
3. What analysis methods will the analyst use to interact with users in gathering data, developing, and testing the system?
4. Who are the key participants? How much time are users willing and able to commit to participating?
5. What are the project deliverables? (What new or updated software, hardware, procedures, and documentation do the users expect to have available for interaction when the project is done?)
6. Who will evaluate the system and how will they evaluate it? What are the steps in the assessment process? How will the results be communicated and to whom?  
7. What is the estimated project timeline? How often will analysts report project milestones?  
8. Who will train the users?  
9. Who will maintain the system?  

The project charter describes in a written document the expected results of the systems project (deliverables) and the time frame for delivery. It essentially becomes a contract between the chief analyst (or project manager) and their analysis team with the organizational users requesting the new system.  

**AVOIDING PROJECT FAILURES**  
The early discussions you have with management and others requesting a project, along with the feasibility studies you do, are usually the best defenses possible against taking on projects that have a high probability of failure. Your training and experience will improve your ability to judge the worthiness of projects and the motivations that prompt others to request projects. If you are part of an in-house systems analysis team, you must keep current with the political climate of the organization as well as with financial and competitive situations.  

You can also learn from the wisdom gained by people involved in earlier project failures. When asked to reflect on why projects had failed, professional programmers cited the setting of impossible or unrealistic dates for completion by management, belief in the myth that simply adding more people to a project would expedite it (even though the original target date on the project was unrealistic), and management behaving unreasonably by forbidding the team to seek professional expertise from outside of the group to help solve specific problems.  

Remember that you are not alone in the decision to begin a project. Although apprised of your team’s recommendations, management will have the final say about whether a proposed project is worthy of further study (that is, further investment of resources). The decision process of your team must be open and stand up to scrutiny from those outside of it. The team members should consider that their reputation and standing in the organization are inseparable from the projects they accept.  

**AGILE DEVELOPMENT**  
The agile approach (also called agile methods or agile development) is an innovative philosophy and methodology comprised of systems development practices, techniques, values, and principles (including XP, extreme programming) intended for use in developing systems in a dynamic way.  

Often posed as an alternative way to develop systems, the agile approach seeks to address common complaints arising over the traditional SDLC approach (for being too time-consuming, focusing on data rather than on humans, and being too costly) by being rapid, iterative, flexible, and participative in responding to changing human information requirements, business conditions, and environments.  

We explore agile methods in greater detail in Chapter 6, but it is very relevant in this chapter, because the agile approach also implies agile project management.  

This section is devoted to agile practices that ensure that the project is completed on schedule. The four variables that a systems developer can control are time, cost, quality, and scope. When these four control variables are properly included in the planning, there exists a state of balance between the resources and the activities needed to complete the project, as noted in Figure 3.17.
The activities of the agile approach are coding, testing, listening, and designing. Coding is, of course, essential in any software project. Testing for functionality, performance, and conformance is mandatory. Listening to the customer and other programmers and analysts is essential. Designing a system that is functional, aesthetic, and maintainable is critically important.

The major difference between agile project management and other more traditional types of project management is that as you listen to what users want, you can figure out how much of each resource is required. In order to balance the project outcomes, the analyst using an agile approach can adjust any of these four resource variables.

The agile philosophy assumes, for example, that if the analyst determines the scope, quality, and amount of time needed to complete the project, the analyst can adjust cost. If the project is running behind schedule, simply increase spending by hiring more people. Alternatively, if the analyst predetermined the amount of time, quality, and cost that is required, the analyst could adjust the scope accordingly. In this case, if the project is running late, the analyst may want to consult with the customer to forego some feature, for example. The following subsection elaborates on adjusting each of the resource control variables.

**RESOURCE TRADE-OFFS USING AN AGILE APPROACH**

Completing all of the activities in the project on time within all of the constraints is admirable, but, as you probably have realized by now, in order to accomplish this, project management is crucial. Managing a project doesn’t mean simply getting all the tasks and resources together. It also means that the analyst is faced with a number of trade-offs. Sometimes cost may be predetermined, at other junctures time may be the most important factor. These resource control variables (time, cost, quality, and scope) are discussed below.

**Time** You need to allow enough time to complete your project. Time, however, is split into many separate pieces. You need time to listen to the customers, time to design, time to code, and time to test.

One of our friends is an owner of a Chinese restaurant. Recently, he found himself short-staffed as one of the members of his reliable crew returned to Hong Kong to get married. The owner placed himself in the kitchen so the food was served on time, but stopped greeting his customers out front in the usual way. He sacrificed the listening activity to achieve another, but in this case he found out it was hurting his business. Customers wanted the attention.

It is the same in systems development. You can create quality software, but fail to listen. You can design a perfect system, but not allow enough time to test it. Time is difficult to manage. If you find yourself running short of time, what do you do?

The agile approach challenges the notion that more time will give you the results you want. Perhaps the customer would prefer that you finish on time rather than extending the deadline to add another feature. Customers, we often find, are happy if some of the functionality is up and running on time. Our experience
shows that often a customer is 80-percent satisfied with the first 20 percent of the functionality. This means that when you complete the final 80 percent of the project, the customer may be only slightly happier than they were after you completed the first 20 percent. The message here is be careful not to extend your deadline. The agile approach insists on finishing on time.

Cost  Cost is the second variable we can consider adjusting. Suppose that the activities of coding, designing, testing, and listening are weighing the project down, and the resources we put into time, scope, and quality are not sufficient, even with a normal amount devoted to cost, to balance the project. Essentially we might be required to contribute more resources that require money to balance the project.

The easiest way to increase spending (and hence costs) is to hire more people. This may appear to be the perfect solution. If we hire more programmers, we’ll finish faster. Right? Not necessarily. Picture hiring two people to repair a roof and increase that number to four. Soon the people are bumping into one another. Furthermore, they need to ask each other what still needs to be done. And if there’s a lightning storm, no one will be working. Going from two to four doesn’t mean it will take half of the time. Consider the required increase in communication and other intangible costs when you are considering hiring more people. Remember that when a person joins a team, they do not know the project or the team. They will slow the original members down, because the original members must devote time to getting new members up to speed.

Overtime doesn’t help much either. It increases the cost, but the productivity doesn’t always follow. Tired programmers are less effective than alert programmers. Tired programmers take a long time to complete a task, and they also make mistakes that are even more time consuming to fix.

Is there anything else we can spend our money on? Perhaps. As you read later chapters you will read about a variety of tools that support analysts and programmers. These tools are often a wise investment. Analysts, for example, use graphical packages such as Microsoft Visio to communicate ideas about the project to others, and CASE tools such as Visible Analyst also help speed up projects.

Even new hardware could be a worthwhile expenditure. Laptops and cell phones improve productivity away from the office. Larger visual displays, Bluetooth-enabled keyboards and mice, and more powerful graphics cards can also increase productivity.

Quality  The third resource control variable is quality. If ideal systems are perfect, why is so much effort placed in maintaining systems? Are we already practicing agile development by sacrificing quality in software development? In Chapter 16 we will see the importance of quality and methods (such as TQM and Six Sigma) that help ensure software quality is high.

The agile philosophy, however, does allow the analyst to adjust this resource, and perhaps put less effort into maintaining quality than otherwise would be expected. Quality can be adjusted both internally and externally. Internal quality involves testing software for factors such as functionality (Does a program do what it is supposed to do?) and conformance (Does the software meet certain conformance standards and is it maintainable?). It usually doesn’t pay to tinker with internal quality.

That leaves us with external quality, or how the customer perceives the system. The customer is interested in performance. Some of the questions a customer may ask are: Does the program act reliably (or do software bugs still exist)? Is the output effective? Does the output reach me on time? Does the software run effortlessly? Is the user interface easy to understand and use?
The extreme philosophy of agile development allows some of the external quality issues to be sacrificed. In order for the system to be released on time, the customer may have to contend with some software bugs. If we want to meet our deadline, the user interface may not be perfect. We can make it better in a follow-up version.

Commercial off-the-shelf software manufacturers do sacrifice quality, and it is debatable whether this is the correct approach. So don’t be surprised when your PC software applications (not to mention your operating system and Web browser) are updated often, if developers are using extreme programming as one of their agile practices.

**Scope** Finally, there is scope. In the agile approach, scope is determined by listening to customers and getting them to write down their stories. Then the stories are examined to see how much can be done in a given time to satisfy the customer. Stories should be brief and easy to grasp. Stories will be described in more detail in Chapter 6, but here is a brief example showing four short stories from an online air travel system. Each story is shown in bold type:

- **Display alternative flights.**
- **Prepare a list of the five cheapest flights.**
- **Offer cheaper alternatives.**
  - *Suggest to the customer that they travel on other days, make weekend stays, take special promotions, or use alternate airports.*
- **Purchase a ticket.**
  - *Allow the customer to purchase a ticket directly using a credit card (check validity).*
  - *Allow the customer to choose his or her seat.*
  - *Direct the customer to a visual display of the airplane and ask the customer to select a seat.*

Ideally, the analyst would be able to determine how much time and money was needed to complete each of these stories and be able to set the level of quality for them as well. It is obvious that this system must not sacrifice quality, or credit card purchases may be invalid or customers may show up at the airport without reservations.

Once again agile practices allow extreme measures, so in order to maintain quality, manage cost, and complete the project on time, the agile analyst may want to adjust the scope of the project. This can be accomplished by agreeing with the customer that one or more of the stories can be delayed until the next version of the software. For example, maybe the functionality of allowing customers to choose their own seats can be put off for another time.

In summary, the agile analyst can control any of the four resource variables of time, cost, quality, and scope. Agility calls for extreme measures and places a great deal of importance on completing a project on time. In doing so, sacrifices must be made and the agile analyst will find out that the trade-offs available involve difficult decisions.

**CORE PRACTICES AND ROLES OF THE AGILE APPROACH**

Now that we have discussed how we can manage systems projects using agile concepts, let’s turn our attention to how to develop and plan agile systems. In this section, we will introduce you to core agile practices that differentiate agile...
development from the SDLC approach and other types of development methodologies. Then we will elaborate on some of the roles that participants in agile development play. Next we will describe how agile projects are planned using a concept called the planning game. Finally, we will address some risks and fears of systems development and how the agile approach addresses these risks.

**Four Core Agile Practices** Balancing resources to complete a project on time is one way to manage a project. The best way to manage, however, is to develop practices that yield outstanding results. The agile movement has developed a set of core practices that have changed the way systems are developed. Following are four extreme practices. As you can see, they are radical compared to what we have discussed so far in project management. Figure 3.18 illustrates the four core practices.

1. **Short release.** In order for agile development to be successful, products must be released quickly. That means that even if the programmers couldn’t get all of the features into a piece of software, the version must be released on schedule. Yes, this is extreme, but customers are pleased because they have a product to use. Any improvements can be made later. This practice is widely used in PC software development and is even more common for cell phones and other mobile devices. In the U.S., even tax software is released early in the tax season before all of the IRS tax laws (and forms) are finalized. The tax software developers know that the customer wants the product as soon as possible.

2. **40-hour work week.** The Silicon Valley model for software development encouraged programmers to live at the office, working around the clock. Not so with the agile movement. Agile development teams purposely endorse a cultural core practice in which the team works intensely together during a typical 40-hour work week. This practice attempts to motivate team members to work intensely at the job, and then to take time off so that when they return to work they are relaxed, less stressed, able to see problems, and less prone to make costly errors and omissions because of ineffectual performance or burnout.

3. **Onsite customer.** Most systems developers claim that the customer is vital to system success, but wind up meeting only once or twice with the customer to determine system requirements. The core practice of the onsite customer goes to the extreme by insisting that an expert in the business should work onsite during the entire development process. This person is active in the process, writing user stories (explained in Chapters 2 and 6), communicating to team members, and helping to set priorities. This works well with adoption of an HCI approach, which is user-centered as well.

4. **Pair programming.** We are very familiar with the concept of systems analyst teams; why not have teams of programmers? No, one programmer doesn’t look
over the other’s shoulder to see whether any errors are made. Rather, pair programming means that two programmers who choose to work together both do the programming, run the tests, and talk to one another about ways to efficiently and effectively get the job done. Working with another programmer helps you clarify your thinking. Pair programming saves time, cuts down on careless thinking, sparks creativity, and is a fun way to program.

Roles for People There are many roles people must play in agile development projects, and some people will even be called on to play multiple roles during the effort. The seven roles include those of programmer, customer, tester, tracker, coach, consultant, and something referred to (only half in jest) as the “big boss.” The seven roles are shown in Figure 3.19.

The programmer is often discussed as the heart of the agile development effort. However, agile programmers are quite different in emphasis than other programmers, because they are asked to be excellent communicators. Communication skills are put into play as soon as a development effort begins, because the agile approach encourages working in programming pairs, in which each person codes, although often there is a junior and senior person on a given task. As an analyst/programmer, you also need to have outstanding technical skills to program well, to be able to refactor, and to be able to unit test the code you have written. In addition you need a willingness to approach the hardest problems with simplicity, learn from others, share in coding and design, and have courage to face any fears of inadequacy or failure you may have in approaching new problems.

The next role we discuss is that of customer. The best way to describe it is that the customer must take on new qualities, some of them very much like a developer, while still retaining the essence of what is needed in the system. The most ideally suited customer for the agile team is someone who will be a user of the system and who possesses business knowledge of it. With that in mind, the customer must learn to write user stories, learn to write functional tests for the applications that the programmers create, and make sound decisions about the essential features of the system and even adjustments in project schedules and delivery dates. Customers also need to show that they have courage in the face of difficult scheduling or functional decisions.
A third role is that of tester on an agile development team. Programmers are called on to do both unit and functional testing of any new coding that has been done. Someone who is a programmer also needs to communicate with the customer about functional tests, run tests regularly, maintain the testing tools, and communicate clearly the results of tests.

Another role on the agile development team is that of tracker. This person tracks the overall progress of the group by estimating the time of its tasks and its overall progress toward its goals. The tracker does estimates, but also provides feedback about the team’s estimates. Were they too low or too high? By what percentage were they incorrect? As a tracker you will be able to tell the team this valuable information, so that their estimates improve in accuracy. Trackers also serve as the team memory, keeping track of all of the functional test scores. Defects that are reported are also tracked, and the name of the programmer who took responsibility for handling that defect is also noted. In addition, you track what test cases were added to address each defect.

We next consider the role of coach, who often is an invisible hand guiding the overall process. Because one of the hallmarks of agile development is that each person accepts responsibility for their actions, a coach may seem unnecessary. However, coaches are critical. For instance, they display a calm demeanor when everyone else on the team is panicking. They shape situations indirectly (most of the time), and only occasionally do they need to assertively pull control away from an errant developer, get them back on track, and then turn over the reins to them again. A good coach keeps reminding the team members about the way they agreed to act when everything first started. A coach might remind a programmer, “You agreed to share ownership of the code,” or “You vowed to take the simplest approach first.” Coaches try to bring out the best qualities in all of the other team members, while remaining in the background most of the time.

The next role we examine is that of consultant. The role of a consulting technical expert is a very odd one. If you are serving as a consultant to the team, they will ask you to solve the problem with them, badgering you all the while to challenge any assumptions that you are glossing over. What agile development teams want out of consultants is to learn how to solve their own problems. As they learn from you they grow confident again, and when you have left them they may or may not use the solution you presented, but be forewarned, this is typical.

The last role we consider for the agile development team is that of big boss. The team expects the big boss to demonstrate confidence in them, exhibit courage to adhere to the basic values and principles they have agreed to, and have the potential to point out a mistake if your team drifts off course. The team will want to keep communicating to you (even small changes from the design, or deviations from other goals). Your task as big boss is to figure out how to keep communication flowing without making it a tidal wave. Above all, you do not want to be an obstacle to anything reasonable the team is trying to do. This is a role that demands complete conviction to the agile approach, and a strong sense that if everyone on the team adheres to their basic values and principles, they will probably come up with something worthwhile.

The Planning Game The entire planning process has been characterized using the idea of a planning game (Beck, 2000, p. 86). The planning game spells out rules that can help formulate the agile development team’s relationship with their business customers. Although the rules form an idea of how you want each party to act during development, they are not meant as a replacement for a relationship. They are a basis for building and maintaining a relationship.
So, we use the metaphor of a game. To that end we talk in terms of the goal of the game, the strategy to pursue, the pieces to move, and the players involved. The goal of the game is to maximize the value of the system produced by the agile team. In order to figure the value, you have to deduct costs of development, and the time, expense, and uncertainty taken on so that the development project could go forward.

The strategy pursued by the agile development team is always one of limiting uncertainty (downplaying risk). To do that they design the simplest solution possible, put the system into production as soon as possible, get feedback from the business customer about what’s working, and adapt their design from there.

Story cards become the pieces in the planning game that briefly describe the task, provide notes, and provide an area for task tracking.

There are two main players in the planning game: the development team and the business customer. Deciding which business group in particular will be the business customer is not always easy, because the agile process is an unusually demanding role for the customer to play. Customers decide what the development team should tackle first. Their decisions will set priorities and check functionalities throughout the process.

How Project Risks Are Handled with the Agile Approach

Up until now our discussion about extreme programming has focused on planning and how to adjust our resources as needed. It is important, however, to note that systems projects can and do have serious problems. Those that are developed using agile methods are not immune to such troubles. In order to illustrate what can go wrong in a project, a systems analyst may want to draw a fishbone diagram (also called a cause-and-effect diagram or an Ishikawa diagram). When you examine Figure 3.20, you will see that it is called a fishbone diagram because it resembles the skeleton of a fish.

The value of fishbone diagrams is to systematically list all of the possible problems that can occur. In the case of the agile approach, it is useful to organize the

![FIGURE 3.20](image_url)

A fishbone diagram may be used to identify all the things that can go wrong in developing a system.
fishbone diagram by listing all of the resource control variables on the top and all of the activities on the bottom. Some problems such as schedule slips might be obvious, but others such as scope creep (the desire to add features after the analyst hears new stories) or developing features with little value are not as obvious.

Many of these problems can be avoided using the agile philosophy. Agile development has very short release cycles, so customers have ample opportunity to influence the design and help increase quality. Schedule slips are kept to a minimum. Pair programming helps maintain quality, reduce turnover, minimize scope creep, and keep defects to a minimum.

Listening and responding to customers’ written and spoken stories minimizes situations in which a business is misunderstood. Having an onsite customer minimizes the chance that a project will be cancelled or the business will change dramatically without the agile team knowing it. In many ways extreme programming reduces potential problems. All in all, it is valuable to identify places where things may go wrong, but it is certainly not productive to fear them.

Many other developmental approaches would blame the analysts for problems due to lack of communication, a tendency toward complexity, or fear. However, the agile approach handles these situations by insisting that analysts and customers share values that enable them to stay away from these problems. There is more detail on these shared values in Chapter 6.

DEVELOPMENTAL PROCESS FOR AN AGILE PROJECT

There are activities and behaviors that shape the way development team members and customers act during the development of an agile project. Two words that characterize a project done with an agile approach are interactive and incremental. By examining Figure 3.21, you can see that there are five distinct stages: exploration,
planning, iterations to the first release, productionizing, and maintenance. Notice that the three red arrows that loop back into the “Iterations” box symbolize incremental changes created through repeated testing and feedback that eventually lead to a stable but evolving system. Also note that there are multiple looping arrows that feed back into the productionizing phase. These symbolize that the pace of iterations is increased after a product is released. The red arrow is shown leaving the maintenance stage and returning to the planning stage, so that there is a continuous feedback loop involving customers and the development team as they agree to alter the evolving system.

During exploration, you will explore your environment, asserting your conviction that the problem can and should be approached with extreme programming, assemble the team, and assess team member skills. This stage will take anywhere from a few weeks (if you already know your team members and technology) to a few months (if everything is new). You also will be actively examining potential technologies needed to build the new system. During this stage you should practice estimating the time needed for a variety of tasks. In exploration, customers also are experimenting with writing user stories. The point is to get the customer to refine a story enough so that you can competently estimate the amount of time it will take to build the solution into the system you are planning. This stage is all about adopting a playful and curious attitude toward the work environment, its problems, technologies, and people.

The next stage of the agile development process is called planning. In contrast to the first stage, planning may only take a few days to accomplish. In this stage you and your customers agree on a date anywhere from two months to half a year from the current date to deliver solutions to their most pressing business problems (you will be addressing the smallest, most valuable set of stories). If your exploration activities were sufficient, this stage should be very short.

The third stage in the agile development process is composed of iterations to the first release. Typically these are iterations (cycles of testing, feedback, and change) of about three weeks in duration. You will be pushing yourself to sketch out the entire architecture of the system, even though it is just in outline or skeletal form. One goal is to run customer-written functional tests at the end of each iteration. During the iterations stage you should also question whether the schedule needs to be altered or whether you are tackling too many stories. Make small rituals out of each successful iteration, involving customers as well as developers. Always celebrate your progress, even if it is small, because this is part of the culture of motivating everyone to work extremely hard on the project.

When all iterations are complete, the system is ready to go to the next stage, known as productionizing. Several activities occur during this phase. In this phase the feedback cycle speeds up so that rather than receiving feedback for an iteration every three weeks, software revisions are being turned around in one week. You may institute daily briefings so everyone knows what everyone else is doing. The product is released in this phase, but may be improved by adding other features. Getting a system into production is an exciting event. Make time to celebrate with your teammates and mark the occasion. One of the watchwords of the agile approach, with which we heartily agree, is that it is supposed to be fun to develop systems!

The last stage we consider is maintenance. Once the system has been released, it needs to be kept running smoothly. New features may be added, riskier customer suggestions may be considered, and team members may be rotated on or off the team. The attitude you take at this point in the developmental process is more conservative than at any other time. You are now in a “keeper of the flame” mode rather than the playful one you experienced during exploration.
The philosophy behind the agile development approach is more than just planning and managing a systems project in extreme ways. It is about values and principles.

**SUMMARY**

The five major project management fundamentals that the systems analyst must handle are (1) project initiation, defining the problem (2) determining project feasibility, (3) activity planning and control, (4) project scheduling, and (5) managing systems analysis team members. Projects may be requested by many different people in the business or by systems analysts themselves. When faced with questions of how businesses can meet their goals and solve systems problems, the analyst creates a problem definition. A problem definition is a formal statement of the problem, including (1) the issues of the present situation, (2) the objectives for each issue, (3) the requirements that must be included in all proposed systems, and (4) the constraints that limit system development.

Selecting a project is a difficult decision, because more projects will be requested than can actually be done. Five important criteria for project selection are (1) that the requested project be backed by management, (2) that it be timed appropriately for a commitment of resources, (3) that it move the business toward attainment of its goals, (4) that it be practical, and (5) that it be important enough to be considered over other possible projects.

If a requested project meets these criteria, a feasibility study of its operational, technical, and economic merits can be done. Through the feasibility study, systems analysts gather data that enable management to decide whether to proceed with a full systems study. Project planning includes the estimation of time required for each of the analyst’s activities, scheduling them, and expediting them if necessary to ensure that a project is completed on time. One technique available to the systems analyst for scheduling tasks is the Gantt chart, which displays activities as bars on a graph.

Another technique, called PERT (for Program Evaluation and Review Techniques), displays activities as arrows on a network. PERT helps the analyst determine the critical path and slack time, which is the information required for effective project control. The timeboxing approach uses an absolute due date for the project, and whatever has been accomplished by that due date is implemented. Function point analysis helps the analyst quantitatively estimate the overall length of the software development efforts.

Computer-based project scheduling using PCs is now practical. In addition, personal information managers can be used by analysts to do planning, create repositories for phone and fax numbers, or even launch other programs. Most PIMs can be synchronized with PIMs in Palm computers and other handheld devices, allowing for excellent portability.

Once a project has been judged feasible, the systems analyst must manage the team members and their activities, time, and resources. Such management is accomplished by communicating with team members. Teams are constantly seeking a balance between working on tasks and maintaining relationships in the team. Tensions arising from attempting to achieve this balance must be addressed. Often two leaders of a team will emerge, a task leader and a socioemotional leader. Members must periodically assess team norms to ensure that the norms are functional rather than dysfunctional for the attainment of team goals.
“I hope everyone you’ve encountered at MRE has treated you well. Here’s a short review of some of the ways you can access our organization through HyperCase. The reception area at MRE contains the key links to the rest of our organization. Perhaps you’ve already discovered these on your own, but I wanted to remind you of them now, because I don’t want to get so engrossed in the rest of our organizational problems that I forget to mention them.

“The telephone on the receptionist’s desk has instructions about how to answer the phone in the rest of the organization. You have my permission to pick up the phone if it is ringing and no one else answers it.

“The empty doorway you see is a link to the next room, which we call the East Atrium. You have probably noticed that all open doorways are links to adjacent rooms. Notice the building map displayed in the reception area. You are free to go to public areas such as the canteen, but as you know, you must have an employee escort you into a private office. You cannot go there on your own.

“By now you have probably noticed the two documents and the computer on the small table in the reception area. The little one is the MRE internal phone directory. Just click on an employee name, and if that person is in, he or she will grant you an interview and a tour of the office. I leave you to your own devices in figuring out what the other document is.

“The computer on the table is on and displays the Web home page for MRE. You should take a look at the corporate page and visit all the links. It tells the story of our company and the people who work here. We’re quite proud of it and have gotten positive feedback about it from visitors.

**FIGURE 3.HC1**
The reception room resembles a typical corporation. While you are in this HyperCase screen, find the directory if you want to visit someone.
"If you have had a chance to interview a few people and see how our company works, I'm sure you are becoming aware of some of the politics involved. We are also worried, though, about more technical issues, such as what constitutes feasibility for a training project and what does not."

**HYPERCASE QUESTIONS**

1. What criteria does the Training Unit use to judge the feasibility of a new project? List them.
2. List any changes or modifications to these criteria that you would recommend.
3. Snowden Evans has asked you to help prepare a proposal for a new project tracking system for the Training Unit. Briefly discuss the technical, economic, and operational feasibility of each alternative for a proposed project tracking system for the Training Unit.
4. Which option would you recommend? Use evidence from HyperCase to support your decision.

Managing ecommerce projects is similar to managing traditional IS projects in a number of ways, but there are four ways in which it departs significantly from these practices. The first is that the data you will be coordinating are scattered all over the organization (which has political ramifications); another is that specialized team members are drawn from across the organization (so organizational politics may also loom); a third is that the ecommerce project manager should be emphasizing strategic integration of ecommerce into all the organization's systems; and the fourth is that security concerns must be managed first when establishing an ecommerce site.

It is important that the systems analysis team set reasonable productivity goals for tangible outputs and process activities. Creating a project charter containing user expectations and analyst deliverables is recommended, since unrealistic management deadlines, adding unneeded personnel to a project that is trying to meet an unrealistic deadline, and not permitting developer teams to seek expert help outside their immediate group, were cited by programmers as reasons projects had failed. Project failures can usually be avoided by examining the motivations for requested projects, as well as your team's motives for recommending or avoiding a particular project.

One alternative to the SDLC is the agile approach. The agile development methodology adopts extreme techniques that can be used to manage projects and keep them on schedule. In the agile approach, the resources that an analyst has available must be balanced against the activities performed.

Agile development is different from other project development processes; it utilizes the practices of short releases, a 40-hour work week, an onsite customer, and pair programming. The seven different roles important in the agile development process are programmer, customer, tester, tracker, coach, consultant, and big boss.

In the agile approach, planning is accomplished by a technique called the planning game, which provides rules for the agile development team to follow when structuring their relationships with a customer. Five broad stages in the agile development process are exploration, planning, iterations to the first release, productionizing, and maintenance.
KEYWORDS AND PHRASES

40-hour work week personal information managers (PIMs)
agile approach PERT diagram
critical path planning phase
ecommerce project management productionizing phase
economic feasibility productivity goals
exploration phase project charter
extreme programming (XP) short release
feasibility impact grid (FIG) socioemotional leader
function point analysis task leader
Gantt chart team motivation
iterations to the first release phase technical feasibility
maintenance phase the planning game
onsite customer
operational feasibility timeboxing
pair programming

REVIEW QUESTIONS

1. What are the five major project fundamentals?
2. List three ways to find out about problems or opportunities that might call for a systems solution.
3. List the five criteria for systems project selection.
4. Examine the feasibility impact grid shown in Figure 3.3. List the corporate objectives that seem to be affected positively by ecommerce systems.
5. Define technical feasibility.
6. Define economic feasibility.
7. Define operational feasibility.
8. When is a two-dimensional Gantt chart more appropriate than a one-dimensional Gantt chart?
9. When is a PERT diagram useful for systems projects?
10. List three advantages of a PERT diagram over a Gantt chart for scheduling systems projects.
11. Define the term critical path.
12. Define the technique of timeboxing.
13. What is function point analysis?
14. Explain how different programming languages affect the time it takes to develop a system.
15. How does a project manager assess the risk of things going wrong and take that into consideration of the time needed to complete the project?
16. List the functions of computer-based project scheduling that are available in common software packages.
17. List the functions of some commonly used personal information manager (PIM) software.
18. List the two types of team leaders.
19. What is meant by a dysfunctional team norm?
20. What is meant by team process?
21. What are three reasons that goal setting seems to motivate systems analysis team members?
22. What are four ways in which ecommerce project management differs from traditional project management?
23. What are three reasons programmers cite for project failure?
24. What elements are contained in a project charter?
25. Name the four resource control variables used in the agile approach.
26. Name the four activities referred to in agile modeling.
27. Describe how control variables are used to balance activities so that agile projects are successful.
28. What are the four core practices of the agile modeling approach that distinguish it from other approaches to development?
29. What are seven roles that must be played during the agile development process?
30. What is the meaning of the phrase “the planning game”?
31. What are the stages in agile development?

PROBLEMS

1. Williwonk’s Chocolates of St. Louis makes an assortment of chocolate candy and candy novelties. The company has six in-city stores, five stores in major metropolitan airports, and a small mail order branch. Williwonk’s has a small, computerized information system that tracks inventory in its plant, helps schedule production, and so on, but this system is not tied directly into any of its retail outlets. The mail order system is handled manually.

Recently, several Williwonk’s stores experienced a rash of complaints from mail order customers that the candy was spoiled upon arrival, that it did not come when promised, or that it never arrived; the company also received several letters complaining that candy in various airports tasted stale. Williwonk’s has been selling a new, low-carb, dietetic form of chocolate made with sugar-free, artificial sweetener. Sales have been brisk, but there have been problems shipping the wrong type of chocolate to an address with a diabetic person. There were a number of complaints and Williwonk’s sent a number of free boxes of chocolate to ease the situation.

Management would like to sell products using the Web but only has a few Web pages with information about the company and an order form that could be printed. Web ordering does not exist. One of the senior executives would like to sell customized chocolates with the name of a person on each piece. Although the production area has assured management that this could be easily done, there is no method to order customized chocolates.

Another senior executive has mentioned that Williwonk’s has partnered with several European chocolate manufacturers and will be importing chocolate from a variety of countries. At present, this must be done over the phone, with email, or by mail. The executive wants an internal Web site that will enable employees to order directly from the partner companies. All this has led a number of managers to request trend analysis. Too much inventory results in stale chocolate, while at other times there is a shortage of a certain kind of chocolate.

Seasonal and holiday variation trends would help Williwonk’s maintain an adequate inventory. The inventory control manager has insisted that all changes must be implemented before the next holiday season. “The time for this to be complete is an absolute due date,” remarked Candy, a senior manager. “Make sure that everything works perfectly before the site goes public,” she continues. “I don’t want any customers receiving the wrong customers!” In addition, the order processing manager has mentioned that the system must be secure.

You had been working for two weeks with Williwonk’s on some minor modifications for its inventory information system when you overheard two
managers discussing these occurrences. List the possible opportunities or problems among them that might lend themselves to systems projects.

2. Where is most of the feedback on problems with Williwonk’s products coming from in Problem 1? How reliable are the sources? Explain in a paragraph.

3. After getting to know them better, you have approached Williwonk’s management people with your ideas on possible systems improvements that could address some of the problems or opportunities given in Problem 1.
   a. In two paragraphs, provide your suggestions for systems projects. Make any realistic assumptions necessary.
   b. Are there any problems or opportunities discussed in Problem 1 that are not suitable? Explain your response.

4. Create a problem definition for the Williwonk’s, as described in Problem 1. Estimate the weights of importance. Include at least one requirement and one constraint.

5. Create a list of user requirements for the problem definition created in Problem 4.

6. Brian F. O’Byrne (“F,” he says, stands for “frozen.”) owns a frozen food company and wants to develop an information system for tracking shipments to warehouses.
   a. Using the data from the table in Figure 3.EX1, draw a Gantt chart to help Brian organize his design project.
   b. When is it appropriate to use a Gantt chart? What are the disadvantages? Explain in a paragraph.

7. In addition to a Gantt chart, you’ve drawn Brian a PERT diagram so that you can communicate the necessity to keep an eye on the critical path. Consult Figure 3.EX2, which was derived from the data from Problem 4. List all paths, and calculate and identify the critical path.

8. Cherry Jones owns a homeopathic medicine company called Faithhealers. She sells vitamins and other relatively nonperishable products for those who want choices regarding alternative medicine. Cherry is developing a new system that
FIGURE 3.EX3

Tasks to be performed during systems development of an order fulfillment system.

<table>
<thead>
<tr>
<th>Description</th>
<th>Task</th>
<th>Must Follow</th>
<th>Time (Weeks)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interview executives</td>
<td>A</td>
<td>None</td>
<td>6</td>
</tr>
<tr>
<td>Interview staff in order fulfillment</td>
<td>B</td>
<td>None</td>
<td>3</td>
</tr>
<tr>
<td>Design input prototype</td>
<td>C</td>
<td>B</td>
<td>2</td>
</tr>
<tr>
<td>Design output prototype</td>
<td>D</td>
<td>A, C</td>
<td>3</td>
</tr>
<tr>
<td>Write use cases</td>
<td>E</td>
<td>A, C</td>
<td>4</td>
</tr>
<tr>
<td>Record staff reactions to prototypes</td>
<td>F</td>
<td>D</td>
<td>2</td>
</tr>
<tr>
<td>Develop system</td>
<td>G</td>
<td>E, F</td>
<td>5</td>
</tr>
<tr>
<td>Write up training manual</td>
<td>H</td>
<td>B, G</td>
<td>3</td>
</tr>
<tr>
<td>Train staff working in order fulfillment</td>
<td>I</td>
<td>H</td>
<td>2</td>
</tr>
</tbody>
</table>

would require her staff to be retrained. Given the information in Figure 3.EX3, make a PERT diagram for her and identify the critical path. If Cherry could find a way to save time on the “write use cases” phase, would it help? Why or why not?

9. Angus McIndoe wants to modernize his popular restaurant by adapting it more closely to his the preferences of his repeat customers. Keeping track of his customers’ likes and dislikes. Information such as where they like to sit, what they like to eat, when they normally arrive at the restaurant are all items of interest to him, since he believes that in this way he can better serve his customers. Angus has asked you to develop a system for him that will help make his customers happy while increasing his business.

You have heard what Angus had to say about his customers. There are certainly more preferences that he can keep track of.

Develop a problem definition for Angus, similar to one developed for Catherine’s Catering in this chapter.

10. Michael Cerveris owns a business that distributes commercial barber and hairdresser equipment to a large group of independently owned shops across the country. He needs a system but is concerned about how long it will take to develop a working one.

Use the following information, along with the example earlier in the book, to calculate the time it would take a team of analysts to develop an operational system.

a. There are three external inputs. Input A references two files and four data elements. Input B references two files and eight data elements. Input C references four files and ten data elements. Using the table in step 1, look up the values for inputs A, B, and C. Are they low, average, or high?

b. There are two external interface files. The first one contains three groups of elements consisting of 50 different data elements. The second contains six groups of elements consisting of 20 different data elements. Using the table in step 2, look up the values for the first and second external interface files. Are they low, average, or high?

c. Using the information in parts a and b, plus the additional information in the table in Figure 3.EX4, calculate the total unadjusted function points using the table in step 3.

FIGURE 3.EX4

Some data to use in function point analysis for the commercial hairdresser equipment problem.

<table>
<thead>
<tr>
<th>Component</th>
<th>Total Number</th>
<th>Low</th>
<th>Average</th>
<th>High</th>
</tr>
</thead>
<tbody>
<tr>
<td>External inputs</td>
<td>See part a.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>External outputs</td>
<td>10</td>
<td>2</td>
<td>7</td>
<td>1</td>
</tr>
<tr>
<td>External queries</td>
<td>18</td>
<td>3</td>
<td>13</td>
<td>2</td>
</tr>
<tr>
<td>Internal logical files</td>
<td>15</td>
<td>3</td>
<td>8</td>
<td>4</td>
</tr>
<tr>
<td>External interface files</td>
<td>See part b.</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
d. Assume the first three systems characteristics had no effect on the complexity of the system, but the remaining 11 had a strong effect (rating = 5) on the complexity. Calculate the value adjustment factor (VAF) using step 4.
e. Calculate the function point count using the formula in step 5.

11. Michael still needs to know how long it will take to complete the system. Taking the result you obtained in the previous problem, tell Michael how many months it would normally take to finish a system using:
a. COBOL.
b. C++ or Java.
c. PowerBuilder.

12. Michael is razor sharp. You have just given him your estimate, but he asks you further questions about risk. He admits three possible problems that might delay the problem. Each delay would set the project back 40 days (eight work weeks). You feel that the risk is moderately low; perhaps 20 percent on the first two and only 10 percent on the third. How much time (in days) should be added to the project timetable for these possible delays?

13. Recently, two analysts just out of college have joined your systems analyst group at the newly formed company, Mega Phone. When talking to you about the group, they mention that some things strike them as odd. One is that group members seem to look up to two group leaders, Bill and Penny, not just one.

Their observation is that Bill seems pretty relaxed, whereas Penny is always planning and scheduling activities. They have also observed that everyone “just seems to know what to do” when they get into a meeting, even though no instructions are given. Finally, they have remarked on the openness of the group in addressing problems as they arise, instead of letting things get out of hand.

a. By way of explanation to the new team members, label the types of leaders Bill and Penny appear to be, respectively.
b. Explain the statement that “everyone just seems to know what to do.” What is guiding their behavior?
c. What concept best describes the openness of the group that the new team members commented on?

14. Prepare a list of activities for a systems development team for an online travel agent that is setting up a Web site for customers. Now suppose you are running out of time. Describe some of your options. Describe what you will trade off to get the Web site released in time.

15. Given the situation for Williwonk’s chocolates (Problem 1), which of the four agile modeling resource variables may be adjusted?

GROUP PROJECTS

1. With your group members, explore project management software such as Microsoft Project. What features are available? Work with your group to list them. Have your group evaluate the usefulness of the software for managing a systems analysis and design team project. In a paragraph, state whether the software you are evaluating facilitates team member communication and management of team activities, time, and resources. State which particular features support these aspects of any project. Note whether the software falls short of these criteria in any regard.

2. Within your group, assign some of the roles that people take on in agile development. Make sure that one person is an onsite customer and at least two people are programmers. Assign other roles, such as coach, as you see fit. Simulate
the systems development situation discussed in Problem 7, or have the person acting as the onsite customer choose an ecommerce business with which they are familiar. Assume that the customer wants to add some functionality to their Web site. Role-play a scenario showing what each person would do if this was being approached through agile methods. Write a paragraph that discusses the constraints that each person faces in enacting his or her role.

SELECTED BIBLIOGRAPHY

Chip enters Anna’s office one day, saying, “I think the project will be a good one, even though it’s taking some long hours to get started.”

Anna looks up from her screen and smiles. “I like what you’ve done in getting us organized,” she says. “I hadn’t realized Visible Analyst could help us this much with project management. I’ve decided to do a PERT diagram for the data gathering portion of the project. It should help us plan our time and work as a team on parallel activities.”

“Can I take a look at the PERT diagram?” asks Chip.

Anna shows him a screen with a PERT diagram on it (see Figure E3.1) and remarks, “This will help immensely. It is much easier than planning haphazardly.”

“I notice that you have Gather Reports, Gather Records and Data Capture Forms, and Gather Qualitative Documents as parallel tasks,” notes Chip, gazing at the screen.

“Yes,” replies Anna. “I thought that we would split up the time that it takes to gather the information. We can also divide up the task of analyzing what we have learned.”

“I notice that you have a rather large number of days allocated for interviewing the users,” notes Chip.

“Yes,” replies Anna. “This activity also includes creating questions, sequencing them, and other tasks, such as taking notes of the office environment and analyzing them. I’ve also assumed a standard of six productive hours per day.”

**Figure E3.1**

A PERT diagram for Central Pacific University that is used for gathering information.
Anna glances at her watch. “But now it’s getting late. I think we’ve made a lot of progress in setting up our project. Let’s call it a day, or should I say evening? Remember, I got us tickets for the football game.”

Chip replies, “I haven’t forgotten. Let me get my coat, and we’ll walk over to the stadium together.”

Walking across campus later, Chip says, “I’m excited. It’s my first game here at CPU. What’s the team mascot, anyway?”

“Chipmunks, of course,” says Anna.

“And the team colors?" Chip asks, as they enter the stadium.

“Blue and white,” Anna replies.

“Oh, that’s why everyone’s yelling, ‘Go Big Blue!’” Chip says, listening to the roar of the crowd.

“Precisely,” says Anna.

EXERCISES

E-1. Use Visible Analyst to view the Gathering Information PERT diagram.

E-2. List all paths and calculate and determine the critical path for the Gathering Information PERT diagram.

E-3. Use Visible Analyst to create the PERT diagram shown in Figure E3.2. It represents the activities involved in interviewing the users and observing their offices.

The exercises preceded by a Web icon indicate value-added material is available from the Web site at www.prenhall.com/kendall. Students can download a sample Visible Analyst Project and a Microsoft Access database that can be used to complete the exercises. Visible Analyst software can be packaged with this text for an additional fee.
E-4. List all paths and calculate and determine the critical path for the Interviewing Users PERT diagram.

E-5. Use Visible Analyst to create a PERT diagram for creating system prototypes. The activity information is shown in Figure E3.3.

<table>
<thead>
<tr>
<th>Activity</th>
<th>Predecessor</th>
<th>Duration</th>
</tr>
</thead>
<tbody>
<tr>
<td>A. Determine overall prototype screens and reports</td>
<td>None</td>
<td>2</td>
</tr>
<tr>
<td>B. Determine report and screen contents</td>
<td>A</td>
<td>4</td>
</tr>
<tr>
<td>C. Create report prototypes</td>
<td>B</td>
<td>3</td>
</tr>
<tr>
<td>D. Create screen prototypes</td>
<td>B</td>
<td>4</td>
</tr>
<tr>
<td>E. Obtain report prototype feedback</td>
<td>C</td>
<td>1</td>
</tr>
<tr>
<td>F. Obtain screen prototype feedback</td>
<td>D</td>
<td>2</td>
</tr>
<tr>
<td>G. Modify report prototypes</td>
<td>E</td>
<td>2</td>
</tr>
<tr>
<td>H. Modify screen prototypes</td>
<td>F</td>
<td>4</td>
</tr>
<tr>
<td>I. Obtain final approval</td>
<td>G, H</td>
<td>2</td>
</tr>
</tbody>
</table>

**FIGURE E3.3**
A list of activities and estimated duration times for the CPU project.