Chapter 2

Succeeding as a Systems Analyst

Learning Objectives

After studying this chapter, you should be able to:

- Discuss the analytical skills, including systems thinking, needed for a systems analyst to be successful.
- Describe the technical skills required of a systems analyst.
- Discuss the management skills required of a systems analyst.
- Identify the interpersonal skills required of a systems analyst.
- Describe the systems analysis profession.

Introduction

In the first chapter, you learned about the different types of information systems developed in organizations, the people who develop them, and the project environment in which systems are developed. Before we explore the systems development life cycle in more detail, however, we need to examine the skills needed to succeed as a systems analyst. You will first examine the analytical skills a systems analyst needs, then discuss the technical, management, and interpersonal skills required of a good analyst. One of the key analytical skills you will study is systems thinking, or the ability to see things as systems. You probably learned about systems and systems thinking in your introductory information systems class, so we will review here the highlights of systems and systems thinking that directly affect the design of information systems and how a systems analyst develops systems.

As illustrated in Figure 2-1, an analyst works throughout all phases of the systems development life cycle. The life cycle model represents the process of developing information systems, the same process you read about in Chapter 1. The skills the analyst needs to be successful are represented by the objects placed in the diagram. The laptop computer represents technical skills; the briefcase represents management skills; the magnifying glass represents analytical skills; and the telephone represents interpersonal skills. As you can see, to follow the guidelines established by any development methodology, an analyst needs to rely on many skills. Although we cannot possibly provide thorough coverage of these skills in this chapter, some will be covered in considerable depth in later chapters while others are discussed more generally. Our goal for these general
skills is to sensitize you to abilities that you need to develop from other courses and materials in order to become a successful systems analyst. The chapter ends by stepping back from these specific skills to examine systems analysis as a profession, with its own standards of practice, ethics, and career paths.

**ANALYTICAL SKILLS FOR SYSTEMS ANALYSTS**

Given the title systems analyst, you might think that analytical skills are the most important. While there is no question that analytical skills are essential, other skills are equally required. First, however, we will focus on the four sets of analytical skills: systems thinking, organizational knowledge, problem identification, and problem analyzing and solving.

**Systems Thinking: A Review**

If you counted the number of times each key term is used in this book, the key term used most frequently would undoubtedly be system. Let’s take the time now to examine systems in general and information systems in particular. (For a more thorough treatment of system concepts, see Martin, et al., 1999). Let’s start by examining what we mean by a system and identify the characteristics that define a system.

**Definitions of a System and Its Parts** A system is an interrelated set of components with an identifiable boundary, working together for some purpose. A system has nine characteristics (see Figure 2-2):

1. Components
2. Interrelated components
3. A boundary
4. A purpose
5. An environment
6. Interfaces
7. Input
8. Output
9. Constraints

A system is made up of components. A component is either an irreducible part or an aggregate of parts, also called a subsystem. The simple concept of a component is very powerful. For example, just as with an automobile or a stereo system with proper design, we can repair or upgrade the system by changing individual components without having to make changes throughout the entire system. The components are interrelated; that is, the function of one is somehow tied to the functions of the others. For example, the work of one component, such as producing a daily report of customer orders received, may not progress successfully until the work of another component is finished, such as sorting customer orders by date of receipt. A system has a boundary, within which all of its components are contained and which establishes the limits of a system, separating the system from other systems. Components within the boundary can be changed whereas things outside the boundary cannot be changed. All of the components work together to achieve some overall purpose for the larger system: the system's reason for existing.

A system exists within an environment—everything outside the system's boundary. For example, we might consider the environment of a state university to include the

Component: An irreducible part or aggregation of parts that make up a system, also called a subsystem.

Interrelated components: Dependence of one subsystem on one or more subsystems.

Boundary: The line that marks the inside and outside of a system and that sets off the system from its environment.

Purpose: The overall goal or function of a system.

Environment: Everything external to a system that interacts with the system.

Figure 2-2
A general depiction of a system
Figure 2-3
Special characteristics of interfaces

INTERFACE FUNCTIONS
Because an interface exists at the point where a system meets its environment, the interface has several special, important functions. An interface provides:

- **Security**, protecting the system from undesirable elements that may want to infiltrate it
- **Filtering**, unwanted data, both for elements leaving the system and entering it
- **Coding and decoding**, incoming and outgoing messages
- **Detecting and correcting errors**, in its interaction with the environment
- **Buffering**, providing a layer of slack between the system and its environment, so that the system and its environment can work on different cycles and at different speeds
- **Summarizing**, raw data and transforming them into the level of detail and format needed throughout the system (for an input interface) or in the environment (for an output interface)

Because interface functions are critical in communication between system components or a system and its environment, interfaces receive much attention in the design of information systems (see Chapters 13 and 14).

**Interface**: Point of contact where a system meets its environment or where subsystems meet each other.

**Constraint**: A limit to what a system can accomplish.

**Input**: Whatever a system takes from its environment in order to fulfill its purpose.

**Output**: Whatever a system returns to its environment in order to fulfill its purpose.

legislature, prospective students, foundations and funding agencies, and the news media. Usually the system interacts with its environment, exchanging, in the case of an information system, data and information. The points at which the system meets its environment are called **interfaces**, and there are also interfaces between subsystems (Figure 2-3 provides a list of functions performed by interfaces). An example of a subsystem interface is the clutch subsystem, which acts as the point of interaction between the engine and transmission subsystems of a car. As can be seen from Figure 2-3, interfaces may include much functionality. You will spend a considerable portion of time in systems development dealing with interfaces, especially interfaces between an automated system and its users (manual systems) and interfaces between different information systems. It is the design of good interfaces that permits different systems to work together without being too dependent on each other.

A system must face **constraints** in its functioning because there are limits (in terms of capacity, speed, or capabilities) to what it can do and how it can achieve its purpose within its environment. Some of these constraints are imposed inside the system (for example, a limited number of staff available) and others are imposed by the environment (for example, due dates or regulations). A system takes **input** from its environment in order to function. Mammals, for example, take in food, oxygen, and water from the environment as input. Finally, a system returns **output** to its environment as a result of its functioning and thus achieves its purpose.

Now that you know the definition of a system and its nine important characteristics, let’s take an example of a system and use it to illustrate the definition and each system characteristic. Consider a system that is familiar to you: a fast-food restaurant (see Figure 2-4).

How is a fast-food restaurant a system? Let’s take a look at the fictional Hoosier Burger Restaurant in Bloomington, Indiana. First, Hoosier Burger has components or subsystems. We can figure out what the subsystems are in many ways but, for the sake of illustration, let’s focus on Hoosier Burger’s physical subsystems as follows: kitchen, dining room, counter, storage, and office. As you might expect, the subsystems are interrelated and work together to prepare food and deliver it to customers, one purpose for the restaurant’s existence. Food is delivered to Hoosier Burger early in the morning, kept in storage, prepared in the kitchen, sold at the counter, and often eaten in the dining room. The boundary of Hoosier Burger is represented by its physical walls and the primary purpose for the restaurant’s existence is to make a profit for its owners, Bob and Thelma Mellankamp.

Hoosier Burger’s environment consists of those external elements that interact with the restaurant, such as customers (many of whom come from nearby Indiana University), the local labor supply, food distributors (much of the produce is grown locally), banks, and neighborhood fast-food competitors. Hoosier Burger has one
interface at the counter where customers place orders and another at the back door where food and supplies are delivered. Still another interface is the telephone managers use regularly to talk with bankers and food distributors. The restaurant faces several constraints. It is designed for the easy and cost-effective preparation of certain popular foods, such as hamburgers and milk shakes, which constrains the restaurant in the foods it may offer for sale. Hoosier Burger's size and its location in the university neighborhood constrain how much money it can make on any given day. The Monroe County Health Department also imposes constraints, such as rules governing food storage. Inputs include, but are not limited to, ingredients for the burgers and other food as well as cash and labor. Outputs include, but are not limited to, prepared food, bank deposits, and trash.

**Important System Concepts** Once we have recognized something as a system and identified the system's characteristics, how do we understand the system? Further, what principles or concepts about systems guide the design of information systems? A key aspect of a system for building systems is the system's relationship with its environment. Some systems, called **open systems**, interact freely with their environments, taking in input and returning output. As the environment changes, an open system must adapt to the changes or suffer the consequences. A **closed system** does not interact with the environment; changes in the environment and adaptability are not issues for a closed system. However, all business information systems are open, and in order to understand a system and its relationships to other information systems, to the organization, and to the larger environment, you must always think of information systems as open and constantly interacting with the environment.

There are several other important systems concepts with which systems analysts need to become familiar:

- Decomposition
- Modularity
- Coupling
- Cohesion

---

**Open system**: A system that interacts freely with its environment, taking input and returning output.

**Closed system**: A system that is cut off from its environment and does not interact with it.
**DECOMPOSITION FUNCTIONS**

Decomposition aids a systems analyst and other systems development project team members by:
- Breaking a system into smaller, more manageable, and understandable subsystems
- Facilitating the focusing of attention on one area (subsystem) at a time without interference from other parts
- Allowing attention to concentrate on the part of the system pertinent to a particular audience, without confusing people with details irrelevant to their interests
- Permitting different parts of the system to be built at independent times and/or by different people

In addition, you need to understand the differences between viewing a system at a logical and at a physical level, each with associated descriptions concentrating on different aspects of a system.

Decomposition deals with being able to break down a system into its components. These components may themselves be systems (subsystems) and can be broken down into their components as well. How does decomposition aid understanding of a system? Decomposition results in smaller and less complex pieces that are easier to understand than larger, complex pieces. Decomposing a system also allows us to focus on one particular part of a system, making it easier to think of how to modify that one part independently of the entire system (Figure 2-5). Figure 2-6 shows the decomposition of a portable compact disc (CD) player. At the highest level of abstraction, this system simply accepts CDs and settings of the volume and tone controls as input and produces music as output. Decomposing the system into subsystems reveals the system’s inner workings: There are separate systems for reading the digital signals from the CDs, for amplifying the signals, for turning the signals into sound waves, and for controlling the volume and tone of the sound. Breaking the subsystems down into their components would reveal even more about the inner workings of the system and greatly enhance our understanding of how the overall system works.

**Modularity** is a direct result of decomposition, refers to dividing a system up into chunks or modules of a relatively uniform size. Modules can represent a system simply, making it not only easier to understand but also easier to redesign and rebuild.

**Coupling** is the extent to which subsystems are dependent on each other. Subsystems should be as independent as possible. If one subsystem fails and other subsystems are highly dependent on it, the others will either fail themselves or have
problems functioning. Looking at Figure 2-6, we would say the components of a portable CD player are tightly coupled. The amplifier and the unit that reads the CD signals are wired together in the same container, and the boundaries between these two subsystems may be difficult to draw clearly. If one subsystem fails, the entire CD player must be sent off for repair. In a home stereo system, the components are loosely coupled since the subsystems, such as the speakers, the amplifier, the receiver, and the CD player, are all physically separate and function independently. For example, if the amplifier in a home stereo system fails, only the amplifier needs to be repaired.

Finally, cohesion is the extent to which a subsystem performs a single function. In biological systems, subsystems tend to be well differentiated and thus very cohesive. In man-made systems, subsystems are not always as cohesive as they should be.

One final key systems concept with which you should be familiar is the difference between logical and physical systems. Any description of a system is abstract since the definition is not the system itself. When we talk about logical and physical systems, we are actually talking about logical and physical system descriptions.

A logical system description portrays the purpose and function of the system without tying the description to any specific physical implementation. For example, in developing a logical description of the portable CD player, we describe the basic components of the player (signal reader, amplifier, speakers, controls) and their relations to each other, focusing on the function of playing CDs using a self-contained, portable unit. We do not specify whether the earphone jack contains aluminum or gold, where we could buy the laser that reads the CDs, or how much the jack or the laser cost to produce.

The physical system description, on the other hand, is a material depiction of the system, a central concern of which is building the system. A physical description of the portable CD player would provide details on the construction of each subunit, such as the design of the laser, the composition of the earphones, and whether the controls feature digital readouts. A systems analyst should deal with function (logical system description) before form (physical system description), just as an architect does for the analysis and design of buildings.

**Benefiting from Systems Thinking** The first step in systems thinking is to be able to identify something as a system. This identification also involves recognizing each of the system’s characteristics, for example, identifying where the boundary lies and all of the relevant inputs. But once you have identified a system, what is the value of thinking of something as a system? Visualizing a set of things and their interrelationships as a system allows you to translate a specific physical situation into more general, abstract terms. From this abstraction, you can think about the essential characteristics of a specific situation. This in turn allows you to gain insights you might never get from focusing too much on the details of the specific situation. Also, you can question assumptions, provide documentation, and manipulate the abstract system without disrupting the real situation.

Let’s look again at Hoosier Burger. How can visualizing a fast-food restaurant as a system help us gain insights about the restaurant that we might not get otherwise? Let’s imagine that Hoosier Burger is facing more demand for its food than it can handle. Some people are convinced that its hamburgers are the best in Bloomington, maybe even in southern Indiana. Many people, especially IU students and faculty, frequently eat at Hoosier Burger, and the staff is having a difficult time keeping up with the demand. For the owner-managers, Bob and Thelma Mellankamp, the high level of demand is both a problem and an opportunity. The problem is that if the restaurant can’t keep up with demand, people will stop coming to eat here, and the owners will lose money. The opportunity is to capitalize on Hoosier Burger’s popularity and serve even more customers every day, making larger profits for the owners (which is the purpose of their system).
How does looking at Hoosier Burger as a system help? By decomposing the restaurant into subsystems, we can analyze each subsystem separately and discover if one or more subsystems is at capacity. Capacity is a general problem common to many systems. Let’s say, after careful study, we discover that the kitchen, storage, and dining room subsystems have plenty of available capacity. However, the counter is unable to handle the rush of people. Customers have to wait in line for several minutes to place and receive their orders. The counter is the restaurant’s bottleneck; thus the capacity of the counter needs to be increased. If we redesign the counter area or the procedures for taking customer orders, then we can increase the counter’s capacity and better match it to the kitchen’s capacity. Customers will have to wait in line less time to place their orders and they will get their food faster. Fewer customers will turn away because of long lines, which should translate into more food sold and higher profits.

There are other aspects of the system we could have examined, such as outputs, inputs, or environmental conditions, but to make the example more clear and concise, we looked only at subsystems. For this particular problem, decomposing Hoosier Burger into its subsystems enabled us to determine its problem with demand. Other problems may have required an examination of all aspects of the restaurant system.

**Applying Systems Thinking to Information Systems** None of the examples of systems we have examined so far in this chapter have been information systems, even though information systems are the focus of this book. There are two reasons why we have looked at other types of systems first. One is so that you will become accustomed to thinking of some of the many different things you encounter daily as systems and realize how useful systems thinking can be. The second is that thinking of organizations as systems is a useful perspective from which to begin developing information systems. Information systems can be seen as subsystems in larger organizational systems, taking input from, and returning output to, their organizational environments.

Let’s examine a simplified version of an information system as a special kind of system. In our fast-food restaurant example, Hoosier Burger uses an information system to take customer orders, send the orders to the kitchen, monitor goods sold and inventory, and generate reports for management. The information system is depicted as a data flow diagram in Figure 2-7 (you will learn how to draw data flow diagrams in Chapter 8).

As the diagram illustrates, Hoosier Burger’s customer order system contains four components or subsystems: Process Customer Food Order, Update Goods Sold File, Update Inventory File, and Produce Management Reports. The arrows in the diagram show how these subsystems are interrelated. For example, the first process produces four outputs: a Kitchen Order, a Receipt, Goods Sold data, and Inventory Data. The later two outputs serve as input for other subsystems. The dotted line illustrates the boundary of the system. Notice that the Customer, the Kitchen, and the Restaurant Manager (Bob Mellankamp) are all considered to be outside the customer order system. The specific purpose of the system is to facilitate customer orders, monitor inventory, and generate reports; the system’s general purpose is to improve the efficiency of the restaurant’s operations.

Since this information system is smaller in scope and purpose than the Hoosier Burger system itself, its environment is also smaller. For our purposes, we can limit the environment to those entities that interact with the system: Customers, the Kitchen, and the Restaurant Manager. Constraints on the system may or may not be apparent from the diagram. For example, the diagram implicitly shows (by omission) that there is no direct data exchange between the customer order system and information systems used by the restaurant’s suppliers; this prevents the system from automatically issuing an order for supplies directly to the suppliers when inventory falls below a certain level. We do not know, however, if any other Hoosier Burger system
A fast-food restaurant's customer order information system depicted in a data flow diagram.

Figure 2-7

The system consists of a computerized cash register that a clerk uses to enter a customer order and return a receipt to the customer. A piece of paper, the kitchen order, is generated from a printer in the restaurant's kitchen. The cash register sends data on the order about goods sold and inventory to a computer in the restaurant's office, where computer files on goods sold and inventory are updated by applications software. Other application software uses data in the Goods Sold and Inventory files to generate and print reports on a laser printer in the office.

On another level of analysis and description, Hoosier Burger's customer order system can be explained using a logical description of an information system that focuses on the flow and transformation of data. The physical system is one possible implementation of the more abstract, logical information system description. For the logical information system description, it is irrelevant whether the customer's order shows up in the kitchen as a piece of paper or as lines of text on a monitor screen. What's important is the information that is sent to Hoosier Burger's kitchen. For every logical information system description, there can be several different physical implementations of it.

The way we draw information systems shows how we think of them as systems. Data flow diagrams clearly illustrate inputs, outputs, system boundaries, the environment, subsystems, and interrelationships. Purpose and constraints are much more
difficult to illustrate and must therefore be documented using other notations. In total, all elements of the logical system description must address all nine characteristics of a system.

**Organizational Knowledge**

As a systems analyst, you will work in organizations. Whether you are an in-house or contract custom software developer, you must understand how organizations work. In addition, you must understand the functions and procedures of the particular organization (or enterprise) you are working for. Furthermore, many of the systems you will build or maintain serve one organizational department and you must understand how that department operates, its purpose, its relationships with other departments and, if applicable, its relationships with customers and suppliers. Table 2-1 lists various kinds of organizational knowledge that a systems analyst must acquire in order to be successful.

**Problem Identification**

What is a problem? Pounds (1969) defines a problem as the difference between an existing situation and a desired situation. For him, the process of identifying problems is the process of defining differences, so problem solving is the process of finding a way to reduce differences. According to Pounds, a manager defines differences by comparing the current situation to the output of a model that predicts what the output should be. For example, at Hoosier Burger, a certain portion of the food ordered from local produce distributors is expected to go bad before it can be used. Comparing a current food spoilage rate of 10 percent to a desired spoilage rate of 5 percent defines a difference and therefore identifies a problem. In this case, Bob Mellankamp has used a model to determine the desired spoilage rate of 5 percent. The particular model used, showing how fast produce ripens after harvesting, typical delivery times, and how long produce will stay fresh in a refrigerator, has come from

<table>
<thead>
<tr>
<th>TABLE 2-1 Selected Areas of Organizational Knowledge for a Systems Analyst</th>
</tr>
</thead>
<tbody>
<tr>
<td>How Work Officially Gets Done in a Particular Organization</td>
</tr>
<tr>
<td>Terminology, abbreviations, and acronyms</td>
</tr>
<tr>
<td>Policies</td>
</tr>
<tr>
<td>Standards and procedures</td>
</tr>
<tr>
<td>Standards of practice</td>
</tr>
<tr>
<td>Formal organization structure</td>
</tr>
<tr>
<td>Job descriptions</td>
</tr>
<tr>
<td>Understanding the Organization's Internal Politics</td>
</tr>
<tr>
<td>Influence and inclinations of key personnel</td>
</tr>
<tr>
<td>Who the experts are in different subject areas</td>
</tr>
<tr>
<td>Critical incidents in the organization's history</td>
</tr>
<tr>
<td>Informal organization structure</td>
</tr>
<tr>
<td>Coalition membership and power structures</td>
</tr>
<tr>
<td>Understanding the Organization's Competitive and Regulatory Environment</td>
</tr>
<tr>
<td>Government regulations</td>
</tr>
<tr>
<td>Competitors, domestic and international</td>
</tr>
<tr>
<td>Products, services, and markets</td>
</tr>
<tr>
<td>Role of technology</td>
</tr>
<tr>
<td>Understanding the Organization's Strategies and Tactics</td>
</tr>
<tr>
<td>Short- and long-term strategy and plans</td>
</tr>
<tr>
<td>Values and mission</td>
</tr>
</tbody>
</table>
research carried out at Purdue University's College of Agriculture. Based on the research, the Mellankamps have set a standard of a 5 percent spoilage rate, with an acceptable variance of 2 percent in either direction. According to this standard, a 5 percent variance between desired and actual is clearly out of line and merits attention. Another model might have indicated that a 10 percent spoilage rate was acceptable. You can see that understanding how managers identify problems is understanding the models they use to define differences.

In order to identify problems that need solving, you must be able to compare the current situation in an organization to the desired situation. You must develop a repertoire of models to define the differences between what is and what ought to be. It is also important that you appreciate the models that information systems users rely on to identify problems. Every functional area of the organization will use different models to find problems; what is helpful in accounting will not necessarily work well in manufacturing. Often you must be able to see problems from a broader perspective. By relying on models from their own particular functional areas, users may not see the real problem from an organizational view.

Problem Analyzing and Solving

Once a problem has been identified, you must analyze the problem and determine how to solve it. Analysis entails finding out more about the problem. Systems analysts learn through experience, with guidance from proven methods, how to get the needed information from people as well as from organizational files and documents. As you seek out additional information, you also begin to formulate alternative solutions to the problem. Devising solutions leads to a search for more information, which in turn leads to improvements in the alternatives. Obviously, such a process could continue indefinitely, but at some point, the alternatives are compared and typically one is chosen as the best solution. Once the analyst, users, and management agree on the general suitability of the solution, they devise a plan for implementing it.

The approach for analyzing and solving problems we describe was formally described by Herbert Simon and colleagues (Simon, 1960). The approach has four phases: intelligence, design, choice, and implementation. During the intelligence phase, all information relevant to the problem is collected. During the design phase, alternatives are formulated and, during choice the best alternative solution is chosen. The solution is put into practice during the implementation phase.

This problem-analysis and -solving approach should be familiar to you: It is essentially the same general process as that described earlier in the systems development life cycle (see Figure 2-8). Simon's intelligence phase corresponds roughly to the first three phases in the life cycle: project identification and selection, project initiation and planning, and analysis. Simon's design phase corresponds to that part of analysis where alternative solutions are formulated. The detailed solution formulation (once the solution is chosen), however, would be performed in the life cycle's latter part of the design phase. Choice of the best solution is made in stages, first at the end of the analysis phase and then during design. In our life cycle model, activities that occur in design, implementation, and maintenance correspond to Simon's implementation phase.

Simon's problem-solving model is a useful one that lends insight into how people solve certain kinds of problems, but there are other factors in organizations that influence how problems are solved. Among these are personal interests, political considerations, and limits in time and cognitive ability that affect how much information people can gather and process. We will say more about these factors in later chapters; now we will turn to an examination of the technical skills required of systems analysts.
Figure 2-8
The systems development life cycle and Simon’s problem-solving model

TECHNICAL SKILLS

Many aspects of your job as a systems analyst are technically oriented. In order to develop computer-based information systems, you must understand how computers, data networks, database management and operating systems, and a host of other technologies work as well as their potential and limitations. Further, you must be technically adept with different notations for representing, or modeling, various aspects of information systems. You need these technical skills not only to perform tasks assigned to you but also to communicate with the other people with whom you work in systems development (see Chapter 1 for a discussion of the roles of various people in systems analysis and design). Rather than develop a single set of technical skills to use throughout your career, you must constantly re-educate yourself about information technology, techniques, and methodologies. These information technology, techniques, and methodologies change quickly, and you must keep up with the changes. You need to understand alternative technologies (like Microsoft Windows, Linux, and UNIX operating environments) as organizational preferences, since choices vary across companies and over time. Versatility, based on a sound understanding of technical concepts rather than specific tools, gives you the flexibility needed for such a changing skill set.

The following activities will help you stay versatile and up-to-date:

- Read trade publications (for example, Computerworld or PCWeek) and books.
- Join professional societies (for example, the Association of Information Technology Professionals or the Association for Computing Machinery) or other clubs and their meetings.
- Attend classes or teach at a local college. Teaching is a wonderful way to force yourself to stay current and to learn from others.
- Attend any courses or training sessions offered by your organization.
• Attend professional conferences, seminars, or trade shows.
• Participate in electronic bulletin boards, news groups, or conferences on local, national, or international networks.
• Regularly browse Websites that focus on industry news, such as CNET. Many trade publications, like *Computerworld*, also have Websites.

Maybe you have seen the cartoon of the person wearing tattered clothes, looking thin, sitting on a park bench feeding the birds. The caption reads, “He was an outstanding systems analyst, but he took a six-month vacation and fell too far behind in his field.” Being a systems analyst working in the systems field requires continuous learning.

Because of the rapid changes that occur in technology, we do not dwell on specifics in this section. For example, when this book was being written, object-oriented database technology was considered new and experimental. It is quite possible, however, that this technology may be popular and widespread when you read this book. In general, you should be as familiar as possible with such families of technologies as

• Microcomputers, workstations, minicomputers, and mainframe computers
• Programming languages
• Operating systems, both for single machines and networks
• Database and file management systems
• Data communication standards and software for local and wide area networks
• Systems development tools and environments (such as form and report generators and graphical interface design tools)
• Web development languages and tools, such as HTML, Cold Fusion, and Microsoft’s Front Page
• Decision support system generators and data analysis tools

as well as modern methods and techniques for describing, modeling, and building systems. How technical you must be will vary by job assignment and where you are in your career. Often, you will be asked to be more technical in the early stages of your career, and then you will assume more managerial responsibilities as you gain experience. We discuss career progression later in this chapter.

**MANAGEMENT SKILLS**

Systems analysis are almost always members of project teams and are frequently asked to lead teams. Management skills are very useful for anyone in a leadership role. As an analyst, you also need to know how to manage your own work and how to use organizational resources in the most productive ways possible. Self-management, then, is an important skill for an analyst. In this section, we describe four categories of management skills: resource, project, risk, and change management.

**Resource Management**

Any organizational worker must know how to obtain and work effectively with organizational resources. A systems analyst must know how to get the most out of a wide range of resources: system documentation, information technology, and money. For an analyst leading a team, the most important resource is people. A team leader must learn how to best utilize the particular talents of other team members. He or she must also be able to delegate responsibility, empowering people to do the tasks they have been assigned.
Resource management includes the following capabilities:

- Predicting resource usage (budgeting)
- Tracking and accounting for resource consumption
- Learning how to use resources effectively
- Evaluating the quality of resources used
- Securing resources from abusive use
- Relinquishing resources when no longer needed and obsoleting resources when they can no longer be useful

**Project Management**

Effectively managing projects is crucial to a systems analyst’s job. Information systems development projects range from one-person projects that take very little time and effort to multi-person, multi-year efforts costing millions of dollars. The goal of project management is to prevent projects from coming in late and going over budget. In addition, project management is designed to help managers keep track of the project’s progress.

Even if you are not a project leader, you will be given responsibilities for parts of a project, or subprojects. In the role of project or subproject manager, you first need to decompose a (sub)project into several independent tasks. The next step is to determine how the tasks are related to each other and who will be responsible for each task. As we will see in Chapter 3, analysts use established tools and techniques to help manage projects. The most important element, however, is managing the people working on the project. Successful analysts motivate people to work together and instill a sense of trust and interdependence among them. Project management extends beyond the organization to any vendors or contractors working on the project.

Often, in today’s development environment, many aspects of a project may be farmed out to various contractors outside the organization. Using independent contractors has many advantages. A particular contractor may be more skilled than internal personnel in a technology or may be less expensive. If a project is short on time, it may also make sense to contract out some parts of a development project to help speed up the overall process. Many times, however, contractors deliver work that is late or of low quality, or that does not meet requirements. If the system requirements are unstable or not well defined, the potential problems with contractors can be exaggerated. For these reasons, it is just as important to manage outside contractors as it is to manage everyone else involved in a project. Two mechanisms that help manage contractors are contracts and relationship managers. Very well-specified contracts that spell out just exactly what is expected and when, and that lay out explicit sanctions for nonperformance, may motivate contractors to perform up to expectations. On the other hand, very explicit contracts may scare off contractors who know that they cannot live up to such a contract’s terms. Relationship managers act as liaisons between your firm and the contractors. By establishing personal relationships with the parties involved, relationship managers may be in a position to sense trouble before it happens and work with both parties toward reasonable settlements.

**Risk Management**

Risk management is the ability to anticipate what might go wrong in a project. Once risks to the project have been identified, you must be able to minimize the likelihood that those risks will actually occur. If minimizing risk is not possible, then you try to minimize the damage that might result. Risk management also includes knowing where to place resources (such as people) where they can do the most good and prioritizing activities to achieve the greatest gain. We discuss a key part of risk management that is carried out during project justification—risk assessment—in Chapter 6.
**Change Management**

Introducing a new or improved information system into an organization is a change process. In general, people do not like change and tend to resist it; therefore, any change in how people perform their work in an organization must be carefully managed. Change management, then, is a very important skill for systems analysts, who are organizational change agents. You must know how to get people to make a smooth transition from one information system to another, giving up their old ways of doing things and accepting new ways. Change management also includes the ability to deal with technical issues related to change, such as obsolescence and reusability. You will learn more about managing the change that accompanies a new information system in Chapter 17.

**INTERPERSONAL SKILLS**

Although, as a systems analyst, you will be working in the technical area of designing and building computer-based information systems, you will also work extensively with all types of people. Perhaps the most important skills you will need to master are interpersonal. In this part of the chapter, we will discuss the various interpersonal skills necessary for successful systems analysis work: communication skills; working alone and with a team; facilitating groups; and managing expectations of users and managers.

**Communication Skills**

The single most important interpersonal skill for an analyst, as well as for any professional, is the ability to communicate clearly and effectively with others. Analysts should be able to successfully communicate with users, other information systems professionals, and management. Analysts must establish a good, open working relationship with clients early in the project and maintain it throughout by communicating effectively.

Communication takes many forms, from written (memos, reports) to verbal (phone calls, face-to-face conversations) to visual (presentation slides, diagrams). The analyst must be able to master as many forms of communication as possible. Oral communication and listening skills are considered by many information systems professionals as the most important communication skills analysts need to succeed. Interviewing skills are not far behind. All types of communication, however, have one thing in common: They improve with experience. The more you practice, the better you get. Some of the specific types of communication we will mention are interviewing and listening, the use of questionnaires, and written and oral presentations.

**Interviewing, Listening, and Questionnaires**

Interviewing is one of the primary ways analysts gather information about an information systems project. Early in a project, you may spend a large amount of time interviewing users about their work and the information they use. There are many ways to effectively interview someone, and becoming a good interviewer takes practice. We will discuss interviewing in more detail in Chapter 7, but it is important to point out now that asking questions is only one part of interviewing. Listening to the answers is just as important, if not more so. Careful listening helps you understand the problem you’re investigating and, many times, the answers to your questions lead to additional questions that may be even more revealing and probing than the questions you prepared before your interview.

Although interviews are very effective ways of communicating with people and obtaining important information from them, interviews can also be very expensive and time-consuming. Because questionnaires provide no direct means by which to
ask follow-up questions, they are generally less effective than interviews. It is possible, however, though time-consuming, to call respondents and ask them follow-up questions. Questionnaires are less expensive to conduct because the questioner does not have to invest the same amount of time and effort to collect the same information using a questionnaire as he or she does in conducting an interview. For example, using a written questionnaire that respondents complete themselves, you could gather the same information from 100 people in one hour that you could collect from only one person in a one-hour interview. In addition, questionnaires have the advantage of being less biased in how the results are interpreted because the questions and answers are standardized. Creating good questionnaires is a skill that comes only with practice and experience. You will learn more about questionnaire design in Chapter 7.

Written and Oral Presentations At many points during the systems development process, you must document the progress of the project and communicate that progress to others. This communication takes the following forms:

- Meeting agenda
- Meeting minutes
- Interview summaries
- Project schedules and descriptions
- Memoranda requesting information, an interview, participation in a project activity, or the status of a project
- Requests for proposal from contractors and vendors

and a host of other documents. This documentation is essential to provide a written, not just oral, history for the project, to convey information clearly, to provide details needed by those who will maintain the system after you are off the project team, and to obtain commitments and approvals at key project milestones.

The larger the organization and the more complicated the systems development project, the more writing you will have to do. You and your team members will have to complete and file a report at the end of each stage of the systems development life cycle. The first report will be the business case for getting approval to start the project. The last report may be an audit of the entire development process. And at each phase, the analysis team will have to document the system as it evolves. To be effective, you need to write both clearly and persuasively.

As there are often many different parties involved in the development of a system, there are many opportunities to inform people of the project's status. Periodic written status reports are one way to keep people informed, but there will also be unscheduled calls for ad hoc reports. Many projects will also involve scheduled and unscheduled oral presentations. Part of oral presentations involves preparing slides, overhead transparencies, or multimedia presentations, including system demonstrations. Another part involves being able to field and answer questions from the audience.

How can you improve your communication skills? We have four simple yet powerful suggestions:

1. Take every opportunity to practice. Speak to a civic organization about trends in computing. Such groups often look for local speakers to present talks on topics of general interest. Conduct a training class on some topic on which you have special expertise. Some people have found participation in Toastmasters, an international organization with local chapters, a very helpful way to improve oral communication skills.

2. Videotape your presentations and do a critical self-appraisal of your skills. You can view videotapes of other speakers and share your assessments with each other.
3. Make use of writing centers located at many colleges as a way to critique your writing.
4. Take classes on business and technical writing from colleges and professional organizations.

**Working Alone and with a Team**

As a systems analyst, you must often work alone on certain aspects of any systems development project. To this end, you must be able to organize and manage your own schedule, commitments, and deadlines. Many people in the organization will depend on your individual performance, yet you are almost always a member of a team and must work with the team toward achieving project goals. As we saw in Chapter 1, working with a team entails a certain amount of give and take. You need to know when to trust the judgment of other team members as well as when to question it. For example, when team members are speaking or acting from their base of experience and expertise, you are more likely to trust their judgment than when they are talking about something beyond their knowledge. For this reason, the analyst leading the team must understand the strengths and weaknesses of the other team members. To work together effectively and to ensure the quality of the group product, the team must establish standards of cooperation and coordination that guide their work (review Table 1-2 for the characteristics of a successful team).

There are several dimensions to the cooperation and coordination that influence team work. Table 2-2 lists the twelve characteristics of a high-performance team (McConnell, 1996). The first characteristic is a shared vision, which allows each team member to have a clear understanding of the project’s objectives. A shared vision helps team members keep their priorities straight and not allow small items of little significance to become overwhelming and distracting. To provide motivation for team members, the vision also needs to present a challenge to team members. The second characteristic, team identity, emerges as team members work together closely and begin to share a common language and sense of humor. Team identity can lead to the synergy of effort only possible when groups work together well.

Shared vision and team identity are important but they alone may not be enough for a team to actually accomplish something. The third characteristic of high-performance teams is how the teams are organized. A result-driven structure is one that depends on clear roles, effective communication systems, means of monitoring individual performance, and decision making based on facts rather than emotions. Choosing the right people for the team is the fourth characteristic. McConnell (1996) reports that team performance may differ by as much as a factor of 5, depend-

---

**TABLE 2-2 Characteristics of a High-Performance Team (McConnell, 1996)**

| 1. Shared, elevated vision or goal    |
| 2. Sense of team identity           |
| 3. Result-driven structure          |
| 4. Competent team members           |
| 5. Commitment to the team           |
| 6. Mutual trust                     |
| 7. Interdependence among team members|
| 8. Effective communication           |
| 9. Sense of autonomy                 |
| 10. Sense of empowerment             |
| 11. Small team size                  |
| 12. High level of enjoyment          |

Adapted with the permission of The Microsoft Press. All rights reserved.
ing only on the skills and attitudes of a team’s members. Although the skills of each team member are important determinants of how well the team will perform, all members must be committed to the team, the fifth characteristic of high-performance teams. A group of the best and brightest individuals, committed only to their own self-interests, cannot outperform a true team of lesser talents who are genuinely committed to each other and to their joint effort.

The next five characteristics of high-performance teams all have to do with how the team members interact with each other. It is very important that team members develop genuine trust for each other. The need for trust is why you see so many teambuilding exercises; for example, an individual falls backwards into the arms of a fellow team member, not knowing if the other person is really there but trusting that he or she will be. Similarly, members of high-performance teams work interdependently, relying on each others’ strengths; develop effective means of communication; give each team member the autonomy to do whatever he or she believes is best for the team and for the project; and empower each team member.

All of these high-performance characteristics seem to work best, according to McConnell (1996), in small teams no larger than eight to ten people. Finally, it is important that teams have fun. Enjoying working together leads to increased team cohesiveness, which has been shown to be a key ingredient of team productivity (Lakhanpal, 1993).

Facilitating Groups

Sometimes you need to interact with a group in order to communicate and receive information. In Chapter 1, we introduced you to the Joint Application Design (JAD) process in which analysts actively work with groups during systems development. Analysts use JAD sessions to gather systems requirements and to conduct design reviews. The assembled group is the most important resource the analyst has access to during a JAD and you must get the most out of that resource; successful group facilitation is one way to do that. In a typical JAD, there is a trained session leader running the show. He or she has been specially trained to facilitate groups, to help them work together, and to help them achieve their common goals. Facilitation necessarily involves a certain amount of neutrality on the part of the facilitator. The facilitator must guide the group without being part of the group and must work to keep the effort on track by ferreting out disagreements and helping the group resolve differences. Obviously, group facilitation requires training. Many organizations that rely on group facilitation train their own facilitators. Figure 2-9 lists some guidelines for running an effective meeting, a task that is fundamental to facilitating groups.

Managing Expectations

Systems development is a change process, and any organizational change is greeted with anticipation and uncertainty by organization members. Organization members will have certain ideas, perhaps based on their hopes and wishes, about what a new information system will be able to do for them; these expectations about the new system can easily run out of control. Ginzberg (1981) found that successfully managing user expectations is related to successful systems implementation. For you to successfully manage expectations, you need to understand the technology and what it can do. You must understand the work flows that the technology will support and how the new system will affect them. More important than understanding, however, is your ability to communicate a realistic picture of the new system and what it will do for users and managers. Managing expectations begins with the development of the business case for the system and extends all the way through training people to use the finished system. You need to educate those who have few expectations as well as temper the optimism of those who expect the new system to perform miracles.
SYSTEMS ANALYSIS AS A PROFESSION

Even though systems analysis is a relatively new field, those in the field have established standards for education, training, certification, and practice. Such standards are required for any profession.

Whether or not systems analysis is a profession is open to debate. Some feel systems analysis is not a profession because it simply has not been around long enough to have established the rigorous standards that define a profession. Others feel that at least some standards are already in place. There are guidelines for college curricula and there are standard ways of analyzing, designing, and implementing systems. Professional societies that systems analysts may join include the Society for Information Management, the Association of Information Technology Professionals, and the Association for Computing Machinery (ACM). There is a Certified Computing Professional (CCP) exam, much like the Certified Public Accountant (CPA) exam, that you can take to prove your competency in the field, although, unlike the CPA certificate, very few jobs and employers in the IS field require you to have the CCP certificate. Codes of ethics to govern behavior also exist. In this section, we will discuss several aspects of a systems analyst’s job: standards of practice, the ACM code of ethics, and career paths for those choosing to become systems analysts.

Standards of Practice

Standard methods or practices of performing systems development are emerging that make systems development less of an art and more of a science. Standards are developed through education and practice and spread as systems analysts move from one organization to another. We will focus here on four standards of practice: an endorsed development methodology, approved development platforms, well-defined roles for people in the development process, and a common language.

There are several different development methodologies now being used in organizations. Although there is no standardization of a single methodology across all organizations, a few prominent methodologies are in common use. An endorsed development methodology lays out specific procedures and techniques to be used during the development process. These standards are central to promoting consistency and reliability in methods across all of an organization’s development projects. Some methodologies are spread through the work of well-known consultants; others are spread through major consulting firms.

Closely associated with endorsed methodologies are approved development platforms. Some methodologies are closely tied to platforms, but other methodologies are...
more adaptable and can work in close accordance with development platforms that exist in the organization, such as database management systems and 4GLs. The point is that organizations, and hence the analysts who work for them, are standardizing around specific platforms, and standards for development emerge from this standardization.

Roles for the various people involved in the development process are also becoming standardized. End users, managers, and analysts are each assigned certain responsibilities for development projects. The training that analysts receive in college, on their first jobs, and during their interactions with other analysts, combine to create a gestalt of the analyst's job. For example, as you study this book and talk about systems development in your class, you are forming certain ideas about what systems analysts do and how systems are developed in organizations. Your ideas are also shaped and reinforced by the other IS courses you take in college. Once you get your first job, you will receive additional training and you will adjust your understanding of systems analysis accordingly. As you gain experience working on projects and interacting with other analysts, who may have been trained at other universities and in other organizations, your ideas will continue to change and grow, but the basic core of what systems analysis means to you will have been established. Many of the experiences you have on the job will reinforce much of what you have already learned about systems analysis. When you leave an organization and go to work elsewhere, you will carry your understanding of systems analysis with you. Over time, as you and other analysts change jobs and move from one organization to another, what it means to be an analyst becomes standardized across organizations, and the standards of practice in the field help define what it means to be an analyst.

Another factor moving the job of the systems analyst toward professionalism is the development of a common language analysts use to talk to each other. Analysts communicate on the job, at meetings of professional societies, and through publications. As analysts develop a special language for communication among themselves, their language becomes standardized. One example is the Unified Modeling Language (UML), which has emerged as a common way to specify and design information systems based on the object-oriented approach (see Chapter 20). Other examples of communication becoming standardized include the widespread use of common programming languages such as COBOL and C and the spread of SQL as the language of choice for data definition and manipulation for relational databases. As their common language develops, analysts become more cohesive as a group—a characteristic of professions.

Ethics

The ACM is a large professional society made up of information system professionals and academics. It has over 85,000 members. Founded in 1947, the ACM is dedicated to promoting information processing as an academic discipline and to encouraging the responsible use of computers in a wide range of applications. Because of its size and membership, it has much influence in the information systems community. The ACM has developed a code of ethics for its members called the “ACM Code of Ethics and Professional Conduct.” The full statement is reproduced in Figure 2-10. The code applies to all ACM members and directly applies to systems analysts.

Note the emphasis in the Code on personal responsibility, on honesty, and on respect for relevant laws. Notice also that compliance with a code of ethics such as this one is voluntary, although article 4.2 calls for, at a minimum, peer pressure for compliance. No one can force an information systems professional to follow these guidelines. However, it is voluntary compliance with the guidelines that makes someone a professional in the first place. Notice that for leaders there is the burden of educating non-IS professionals about computing—about what computing can and
Association for Computing Machinery Professional Code of Ethics

Preamble
Commitment to ethical professional conduct is expected of every member (voting members, associate members, and student members) of the Association for Computing Machinery (ACM).

This Code, consisting of 24 imperatives formulated as statements of personal responsibility, identifies the elements of such a commitment. It contains many, but not all, issues professionals are likely to face. Section 1 outlines fundamental ethical considerations, while Section 2 addresses additional, more specific considerations of professional conduct. Statements in Section 3 pertain more specifically to individuals who have a leadership role, whether in the workplace or in a volunteer capacity such as with organizations like ACM. Principles involving compliance with this Code are given in Section 4.

(1.0) General Moral Imperatives
(As an ACM member I will . . .)
(1.1) Contribute to society and human well-being.
(1.2) Avoid harm to others.
(1.3) Be honest and trustworthy.
(1.4) Be fair and take action not to discriminate.
(1.5) Honor property rights including copyrights and patent.
(1.6) Give proper credit for intellectual property.
(1.7) Respect the privacy of others.
(1.8) Honor confidentiality.

(2.0) More Specific Professional Responsibilities
(As an ACM computing professional I will . . .)
(2.1) Strive to achieve the highest quality, effectiveness and dignity in both the process and products of professional work.
(2.2) Acquire and maintain professional competence.
(2.3) Know and respect existing laws pertaining to professional work.
(2.4) Accept and provide appropriate professional review.
(2.5) Give comprehensive and thorough evaluations of computer systems and their impacts, including analysis of possible risks.
(2.6) Honor contracts, agreements, and assigned responsibilities.
(2.7) Improve public understanding of computing and its consequences.
(2.8) Access computing and communication resources only when authorized to do so.

(3.0) Organizational Leadership Imperatives
(As an ACM member and an organizational leader I will . . .)
(3.1) Articulate social responsibilities of members of an organizational unit and encourage full acceptance of those responsibilities.
(3.2) Manage personnel and resources to design and build information systems that enhance the quality of working life.
(3.3) Acknowledge and support proper and authorized uses of an organization’s computing and communication resources.
(3.4) Ensure that users and those who will be affected by a system have their needs clearly articulated during the assessment and design of requirements; later the system must be validated to meet requirements.
(3.5) Articulate and support policies that protect the dignity of users and others affected by a computing system.
(3.6) Create opportunities for members of the organization to learn the principles and limitations of computer systems.

(4.0) Compliance with the Code
(As an ACM member I will . . .)
(4.1) Uphold and promote the principles of this Code.
(4.2) Treat violations of this code as inconsistent with membership in the ACM.

Figure 2-10
ACM Code of Ethics and Professional Conduct, Revision Draft No. 19 (9/19/91).
(Copyright © Association for Computing Machinery, reprinted with permission.)
cannot do. The Code also expresses concern for the quality of work life and for protecting the dignity and privacy of others when performing professional work, such as developing information systems.

Though not written specifically for systems analysts, the ACM Code of Ethics can easily be adapted to the systems analysis job. Many systems development projects deal directly with many of the issues addressed in the Code: privacy, quality of work life, user participation, and managing expectations. When an analyst must confront one or more of these issues, the Code can be used as a guide for professional conduct.

It is also important to remember that systems analysts work within organizations. Codes of ethics, such as that approved by the ACM, may not provide all of the guidance analysts need for dealing with ethically questionable situations in business organizations. The study of ethics, however, is a complex and sometimes bewildering exercise, so simplified and targeted approaches can be very helpful. One such approach has been developed by Smith and Hasnas (1999) for business managers, but it can be usefully applied by information systems professionals as well.

Smith and Hasnas describe three different ways to view business problems with ethical considerations. The first is the stockholder approach, which holds that any action taken by a business is ethically acceptable as long as it is legal, not deceptive, and maximizes profits for stockholders.

The second view is the stakeholder approach. A stakeholder is not the same as a stockholder. A stakeholder, like a stockholder, may own part of the firm, but a stakeholder typically has a greater involvement with the firm than does a stockholder. A stakeholder is either vital to the ongoing operation of the firm or is vitally affected by the actions of the firm. According to the stakeholder approach, you first have to determine who your stakeholders are. Then every action you are considering that violates the rights of any one of these stakeholders must be rejected. Only actions that best balance the rights of the different stakeholder groups can be taken by the firm.

The third approach is called the social contract approach. The focus of this approach is much broader than the other two, as it extends beyond stockholders and stakeholders to members of society at large. Any actions, potentially taken by the firm, that are deceptive, that could dehumanize employees, or that could discriminate, must be rejected outright. Further, any potential actions that could reduce the welfare of the members of society must also be eliminated. Only then can actions that would enhance the financial liability of the firm be considered. Table 2-3 lists the key ethical obligations of anyone employing one of these three approaches.

The best way to compare these three approaches is to look at how they could be applied to a business situation. Smith and Hasnas (1999) supply just such an exam-

<p>| Table 2-3 Comparison of Ethical Obligations for Three Different Approaches to Business Ethics (from Smith and Hasnas, 1999). Adapted with permission of MIS Quarterly. |
|---------------------------------|---------------------------------|---------------------------------|</p>
<table>
<thead>
<tr>
<th>Stockholder</th>
<th>Stakeholder</th>
<th>Social Contract</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Conform to laws and regulations</td>
<td>• Determine who are relevant stakeholders</td>
<td>• Reject actions that are fraudulent/deceptive, dehumanize employees, or involve discrimination</td>
</tr>
<tr>
<td>• Avoid fraud and deception</td>
<td>• Determine rights of each; reject options that violate these</td>
<td>• Eliminate options that reduce welfare of society's members</td>
</tr>
<tr>
<td>• Maximize profits</td>
<td>• Accept remaining option that best balances interests of stakeholders</td>
<td>• Choose remaining option that maximizes probability of financial success</td>
</tr>
</tbody>
</table>
ple. In December 1990, Blockbuster Entertainment Corporation announced a plan to sell the rental history of its customers to direct marketers. (Federal law prohibits the disclosure of the titles of videos that people rent, but it is legal to disclose the categories of videos that one rents.) Blockbuster scuttled the plan in response to public outcry. However, the Blockbuster case provides a good example for comparing the three approaches to business ethics.

Under the stockholder approach, Smith and Hasnas say the Blockbuster plan would have been ethically acceptable. It was legal and not deceptive, and the income from selling rental histories would have added to Blockbuster’s profits. Under the stakeholder approach, the different stakeholder groups would have to be identified first, and then the effects of the plan on each group would have to be determined and compared to the effects on other groups. If, for example, the plan resulted in limited income and severely inconvenienced customers, say, for example, in terms of all the junk mail they would receive from the direct marketers who purchased their rental histories, then the plan would probably not be ethically acceptable. However, if the plan generated lots of new revenue and customers were only slightly inconvenienced, then the plan might well be ethically acceptable. Finally, using the social contract approach, the plan to sell rental histories would probably not be ethically acceptable. The primary reason for this determination, according to Smith and Hasnas (1999), is that neither Blockbuster employees nor customers would benefit in any material way from the implementation of the plan.

Although this example is greatly simplified, it does illustrate well the types of ethical dilemmas confronting information technology professionals. It also does a very good job in showing how the ethics of a situation depends greatly on the ethical approach taken to examine the issue in the first place.

**Career Paths**

Currently, there are many different opportunities for a recent college graduate with a degree in management information systems (MIS). Traditionally, most recent graduates took jobs as systems analysts or programmer/analysts with large consulting firms. Many still do. But the traditional path has changed in the past few years. We explore below some of the many alternatives available to information technology specialists, but first we discuss the consulting option, which remains viable and attractive.

The information systems business at consulting firms has been growing at double-digit levels over the past decade, creating a need for hundreds if not thousands of MIS graduates. Typically, if you are hired by a consulting firm, the first thing you do is to report for extensive and intensive training in the tools and technologies the firm uses. Most of these firms have their own campuses where they train new recruits. After training, you would be assigned to a project. The project may or may not be close to the city you have chosen as your base.

As a junior consultant, your job would involve lots of travel, and you would be involved in many projects over the years. You would be exposed to many different organizations, technologies, industries, and systems. One type of system consulting firms have been heavily involved with lately has been enterprise resource planning or ERP systems. These systems are so large and complex, organizations seeking to implement them need the expert help only experienced consulting firms can provide. Once you had been working for the firm for a while, if you were successful, you would have to decide if you wanted to compete for a partner position. Unlike corporations, where stockholders own the company, most consulting firms are organized as partnerships, where the partners own the company. As you might imagine, there are not many partnerships available, and competition for them is fierce. Many consulting firm employees decide not to compete for a partner position. Instead, many go to work for client firms, or start their own small consulting firms.
Another opportunity available to you is to work in the information systems shop of a corporation. The work is very similar to what you would do as an analyst for a consulting firm, except that your clients all work for the same corporation as you do. This does not mean you won’t have to travel. Large corporations have plants and offices all over the world, so you might have to travel to these distant locations as part of your job. You would not be exposed to as many different types of systems and technologies as would be the case if you worked for a consulting firm, but instead you would have the chance to gain deep expertise in the technologies the corporation has chosen to use. You would also have the opportunity, if you chose, to become a division or department manager in the information systems shop. You might also decide you wanted to become the corporation’s Chief Information Officer or CIO.

Until very recently, most corporations chose not to hire new graduates right out of school for their information systems shops, but that is no longer the case for many organizations.

Finally, considering that so many systems organizations use application packages developed by software vendors, there are many opportunities available for the recent MIS graduate in the software industry. The software industry is a massive, multi-billion dollar a year industry, so the range of opportunities is vast, ranging from a job with a large established firm like Microsoft, to working for an Internet start-up. Working for a software vendor, you would work on developing and testing information systems, just as would be the case in a consulting firm or corporation, but unlike those jobs, you would rarely ever see or talk with the end user of the system you develop. Opportunities exist for you to move up in the company or to use what you have learned to start your own software development firm.

Not all recent MIS graduates become systems analysts, however. There are many other types of information technology jobs that make use of the skills you will acquire as part of completing an MIS degree. Among the other opportunities now available for recent MIS graduates are the following:

- Network administration, which involves installing, managing, monitoring, and upgrading the firm’s internal data and communication networks
- Technical support specialist, which involves troubleshooting, customer service, hardware and/or software installation, and systems maintenance (ITAA, 2000)
- Help desk support, in which you attempt to solve user problems and answer user questions about systems they rely on
- E-business and multimedia product and service development, where you help migrate existing systems to the Internet as well as develop new applications that take advantage of trends in electronic business
- Decision support analyst, in which you design database queries and data analysis routines to support business analysis and decision making, often for one department, such as market research or investments
- Data warehouse specialist, which involves converting massive amounts of historical data to aggregated data useful for decision support
- Quality assurance specialist, in which you review and test software to make sure it is as error-free as possible.

Obviously, not all of these opportunities will exist in every company that uses information technology. Not every firm will have need for data warehouse specialists or systems analysts. Yet almost every firm needs network administrators and technical support specialists. These positions account for half of the new information technology jobs created in 2000 (ITAA, 2000). Even though every firm does not employ analysts or database developers, 20 percent of the new jobs created in 2000 fall into these categories. An additional 13 percent of new information technology jobs are related
to E-business and Internet development. In short, the opportunities are widespread, regardless of which particular area of the information technology profession you decide to pursue.

Summary

In this chapter, we have surveyed the skills necessary for success as a systems analyst. The requisite skills are analytical, technical, management, and interpersonal. Analytical skills include the concept of systems thinking, which is one of the most important skills an analyst can learn. Systems thinking provides a disciplined foundation on which all other analyst skills can build. In addition, an analyst needs to understand the nature of business and of the particular enterprise he or she serves and to be able to identify, analyze, and solve problems.

Technical skills change over time as technology changes and analysts need to keep current with changing information technology. This can be accomplished through reading trade journals, joining professional societies, attending or teaching classes, attending conferences, and participating in electronic bulletin boards and news groups. Some technology areas that play a continuing important role are programming languages, operating systems, database management systems, data communications, and systems development techniques and tools.

A useful skill is the ability to manage resources, projects, risk, and change. Interpersonal skills, especially clear communication, are also important. Analysts communicate with team members in interviews, through written and oral presentations, and through facilitating groups. A key component of communicating about information systems is managing the expectations of both users and managers.

The chapter concluded with an examination of the system analyst's position, the standards of practice, the ACM Code of Ethics, and possible career paths. Systems analysis is becoming more of a science and less of an art as the systems analysis field becomes a profession.

Key Terms

1. Boundary
2. Closed system
3. Cohesion
4. Components
5. Constraints
6. Coupling
7. Environment
8. Input
9. Interface
10. Interrelated components
11. Logical system description
12. Modularity
13. Open system
14. Output
15. Physical system description
16. Purpose
17. System

Match each of the key terms above with the definition that best fits it.

- A system that is cut off from its environment and does not interact with it.
- An interrelated set of components, with an identifiable boundary, working together for some purpose.
- An irreducible part or aggregation of parts that make up a system, also called a subsystem.
- Dependence of one part of the system on one or more other system parts.
- The line that marks the inside and outside of a system, and that sets off the system from its environment.
- The overall goal or function of a system.
- Whatever a system returns to its environment in order to fulfill its purpose.
- Everything external to a system that interacts with the system.

- Point of contact where a system meets its environment or where subsystems meet each other.
- A limit to what a system can accomplish.
- Dividing a system up into chunks or modules of a relatively uniform size.
- The extent to which subsystems depend on each other.
- The extent to which a system or subsystem performs a single function.
- Whatever a system takes from its environment in order to fulfill its purpose.
- Description of a system that focuses on the system's function and purpose without regard to how the system will be physically implemented.
- A system that interacts freely with its environment, taking input and returning output.
- Description of a system that focuses on how the system will be materially constructed.