Unit-3

System Analysis

Analysis is the **first systems development life cycle (SDLC) phase where you begin to understand, in depth, the need for system changes**. Systems analysis involves a substantial amount of effort and cost, and is therefore undertaken only after management has decided that the systems development project under consideration has merit and should be pursued through this phase. Most observers would agree that many of the errors in developed systems are directly traceable to inadequate efforts in the analysis and design phases of the life cycle.

The purpose of analysis is to determine what information and information processing services are needed to support selected objectives and functions of the organization. Gathering this information is called **requirements determination**.

The results of the requirements determination can be structured according to three essential views of the current and replacement information systems:

Process: The sequence of data movement and handling operations within the system.

Logic and timing: The rules by which data are transformed and manipulated and an indication of what triggers data transformation.

Data: The inherent structure of data independent of how or when they are processed.

Determining System Requirements

Introduction: Systems analysis is the part of the systems development life cycle in which you determine how the current information system functions and assess what users would like to see in a new system.Analysis has two sub phases:

requirements determination and

requirements structuring.

Techniques used in requirements determination have evolved over time to become more structured and increasingly rely on computer support. We will first study the more **traditional requirements determination** methods, including **interviewing, observing users** in their work environment, and **collecting procedures and other written documents.** We will then discuss more current methods for collecting system requirements.

The first of these methods is **Joint Application Design(JAD)**. Next, you will read about how analysts rely more and more on information systems to help them perform analysis. As you will see, CASE tools, discussed in Chapter 1, are useful in requirements determination, and prototyping has become a key tool for some requirements determination efforts. Finally, you will learn how requirements analysis continues to be an important part of systems analysis and design, whether the approach involves business process redesign, new Agile techniques (such as constant user involvement or usage-centered design), or focuses on developing Internet applications.

Performing Requirements Determination

As shown in Figure 6-1, there are two subphases to systems analysis:

requirements determination and

requirements structuring.

We will address these as separate steps, but you should consider the steps as parallel and iterative. For example, as you determine some aspects of the current and desired system(s), you begin to structure these requirements or build prototypes to show users how a system might behave.

Inconsistencies and deficiencies discovered through structuring and prototyping lead you to explore further the operation of current system(s) and the future needs of the organization. Eventually, your ideas and discoveries converge in a thorough and accurate depiction (the way that something is represented or shown) of current operations and requirements for the new system.

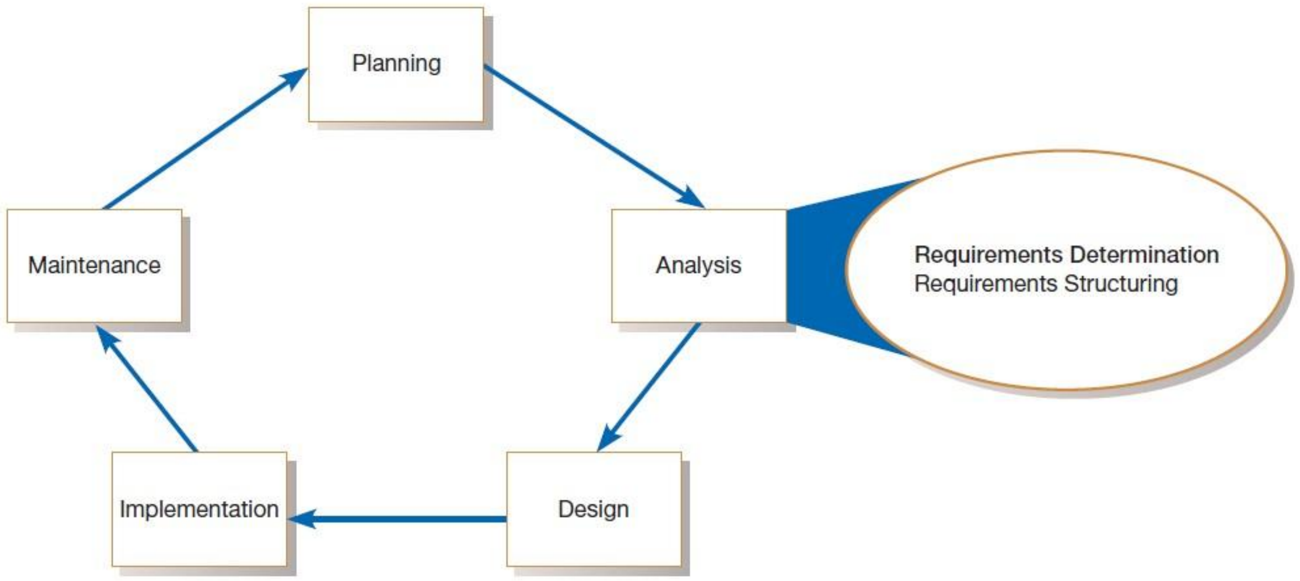


FIGURE 6-1
Systems development life cycle with analysis phase highlighted.

The Process Of Determining Requirements

Once management has granted permission to pursue (follow or start)development of a new system (this was done at the end of the project identification and selection phase of the SDLC) and a project is initiated and planned, you **begin determining what the new system should do**. During requirements determination, you and other analysts gather information on what the system should do from as many sources as possible: from users of the current system; from observing users; and from reports, forms, and procedures. All of the system requirements are carefully documented and prepared for structuring. In many ways, gathering system requirements is like conducting any investigation.

These characteristics include the following:

Impertinence: You should question everything. You need to ask questions such as:

Are all transactions processed the same way?

Could anyone be charged something other than the standard price?

Might we someday want to allow and encourage employees to work for more than one department?

Impartiality: Your role is **to find the best solution to a business problem** or opportunity. It is not, for example, to find a way to justify the purchase of new hardware or to insist on incorporating what users think they want into the new system requirements. You must consider issues raised by allparties and try to **find the best organizational solution**.

Relax constraints: Assume that anything is possible and eliminate the infeasible. For example, do not accept this statement: "**We've always done it that way**, so we have to continue the practice." Traditions are different from rules and policies. Traditions probably started for a good reason but, as the organization and its environment change, they may turn into habits rather than sensible procedures.

Attention to details: Every fact must fit with every other fact. One element out of place means that even the best system will fail at some time. For example, an imprecise (not accurate or exact) definition of who a customer is may mean that you purge (remove) customer data when a customer has no active orders, yet these past customers may be vital contacts for future sales.

Reframing: Analysis is, in part, a creative process. You must challenge yourself to look at the organization in new ways. You must consider how each user views his or her requirements. You must be careful not to jump to the following conclusion: "**I worked on a system like that once—this new system must work the same way as the one I built before.**"

Deliverables and outcomes: The primary deliverables from requirements determination are the **various forms of information** gathered during the determination process:
transcripts of interviews;
notes from observation and analysis of documents;
sets of forms, reports, job descriptions,
other documents; and
computer-generated output such as system prototypes.
In short, anything that the analysis team collects as part of determining system requirements is included in the deliverables resulting from this subphase of the systems development life cycle. Table 6-1 lists examples of some specific information that mightbe gathered during requirements determination.

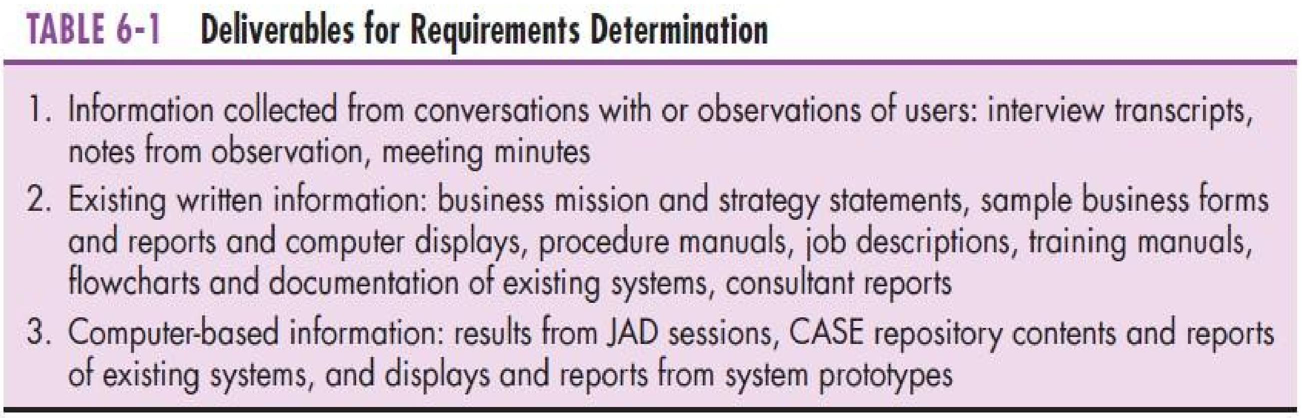


TABLE 6-1 Deliverables for Requirements Determination

1. Information collected from conversations with or observations of users: interview transcripts, notes from observation, meeting minutes
2. Existing written information: business mission and strategy statements, sample business forms and reports and computer displays, procedure manuals, job descriptions, training manuals, flowcharts and documentation of existing systems, consultant reports
3. Computer-based information: results from JAD sessions, CASE repository contents and reports of existing systems, and displays and reports from system prototypes

We could group all this information in three groups:

- **Information collected from conversations with or observations of users** (interview transcripts, notes from observations etc.)
- **Existing written informations** (business mission or strategy, forms, reports, training manuals, flow charts and documentation of existing system etc.)
- **Computer-based information** (results from JRP sessions, CASE repository contents)

Too much analysis is not productive, and the term **analysis paralysis** hasbeen coined (invented) to describe a systems development project that has become bogged down (to be/become so involved in something difficult or complicated that you cannot do anything else) in an abundance of analysis work. Because of the dangers of excessive analysis, today's systems analysts focus more on the system to be developed than on the current system. The techniques you will learn about later in this chapter, **JAD and prototyping**, were developed to keep the analysis effort at a minimum yet still keep it effective. Newer techniques have also been developed to keep requirements determination fast and flexible, including continual user involvement, usage centered design, and the **Planning Game from eXtreme Programming.**

Traditional methods for determining requirements

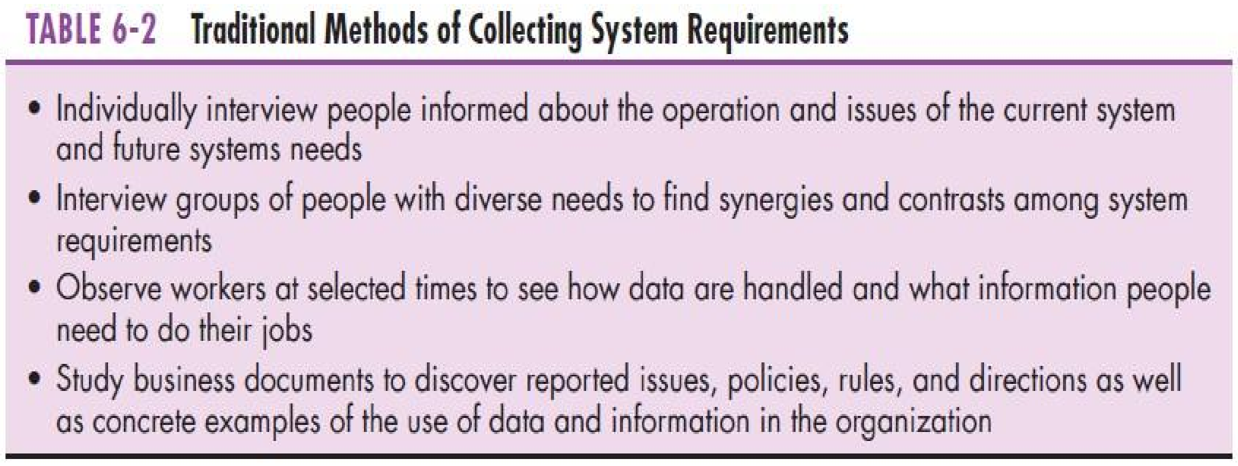


TABLE 6-2 Traditional Methods of Collecting System Requirements

- Individually interview people informed about the operation and issues of the current system and future systems needs
- Interview groups of people with diverse needs to find synergies and contrasts among system requirements
- Observe workers at selected times to see how data are handled and what information people need to do their jobs
- Study business documents to discover reported issues, policies, rules, and directions as well as concrete examples of the use of data and information in the organization

At the core of systems analysis is the collection of information. At the outset, you must collect information about the information systems that are currently being used and how users would like to improve the current systems and organizational operations with new or replacement information systems. One of the best ways to get this information is to talk to the people who are directly or indirectly involved in the different parts of the organizations affected by the possible system changes: **users, managers, funders, and so on**. Another way to find out about the current system is to gather copies of documentation relevant to current systems and business processes.

In this chapter, you will learn about various ways to get information directly from **stakeholders interviews**, **group interviews**, **the Nominal Group Technique, and direct observation.** You will learn about collecting documentation on the current system and organizational operation in the form of written procedures, **forms, reports, and other hard copy.** These traditional methods of collecting system requirements are listed in aboveTable 6-2.

Interviewing and listening

Interviewing is one of the primary ways analysts gather information about an information systems project. Early in a project, an analyst may spend a large amount of time interviewing people about their work, the information they use to do it, and the types of information processing that might supplement their work. Other stakeholders are interviewed to understand **organizational direction, policies, expectations** managers have on the units they supervise, and other non routine aspects of organizational operations. During interviewing you will gather **facts, opinions, and speculation**(Guess, when you guess possible answers to a question without having enough information to be certain) and observe body language, emotions, and other signs of what people want and how they assess current systems. There are **many ways to effectively interview someone**, and no one method is necessarily better than another. Some guidelines you should keep in mind when you interview, summarized in Table 6-3, are discussed next.

First, **you should prepare thoroughly before the interview**. Set up an **appointment at a time and for a duration convenient for the interviewee.** The general nature of the interview should be explained to the interviewee in advance. You may **ask the interviewee to think about specific questions or issues or to review certain documentation to prepare for the interview**. You should spend some time thinking about what you need to find out and write down your questions. Do not assume that you can anticipate(expect) all possible questions. You want the interview to be natural, and to some degree, you want to spontaneously direct the interview as you discover what expertise the interviewee brings to the session.

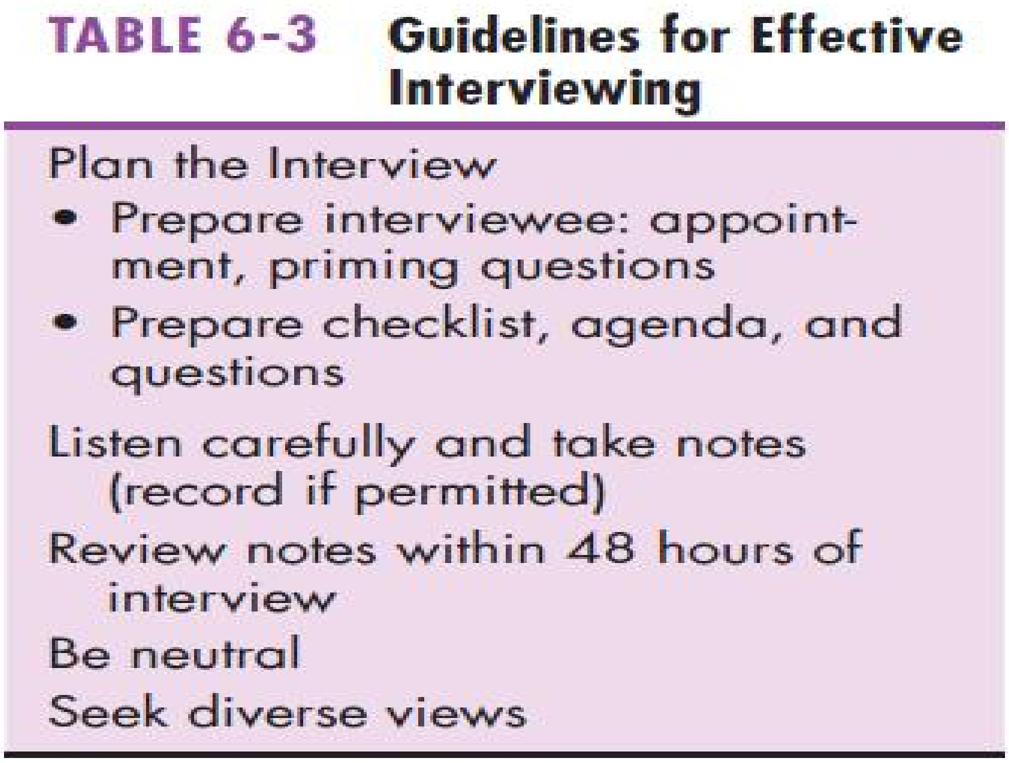


TABLE 6-3 Guidelines for Effective Interviewing

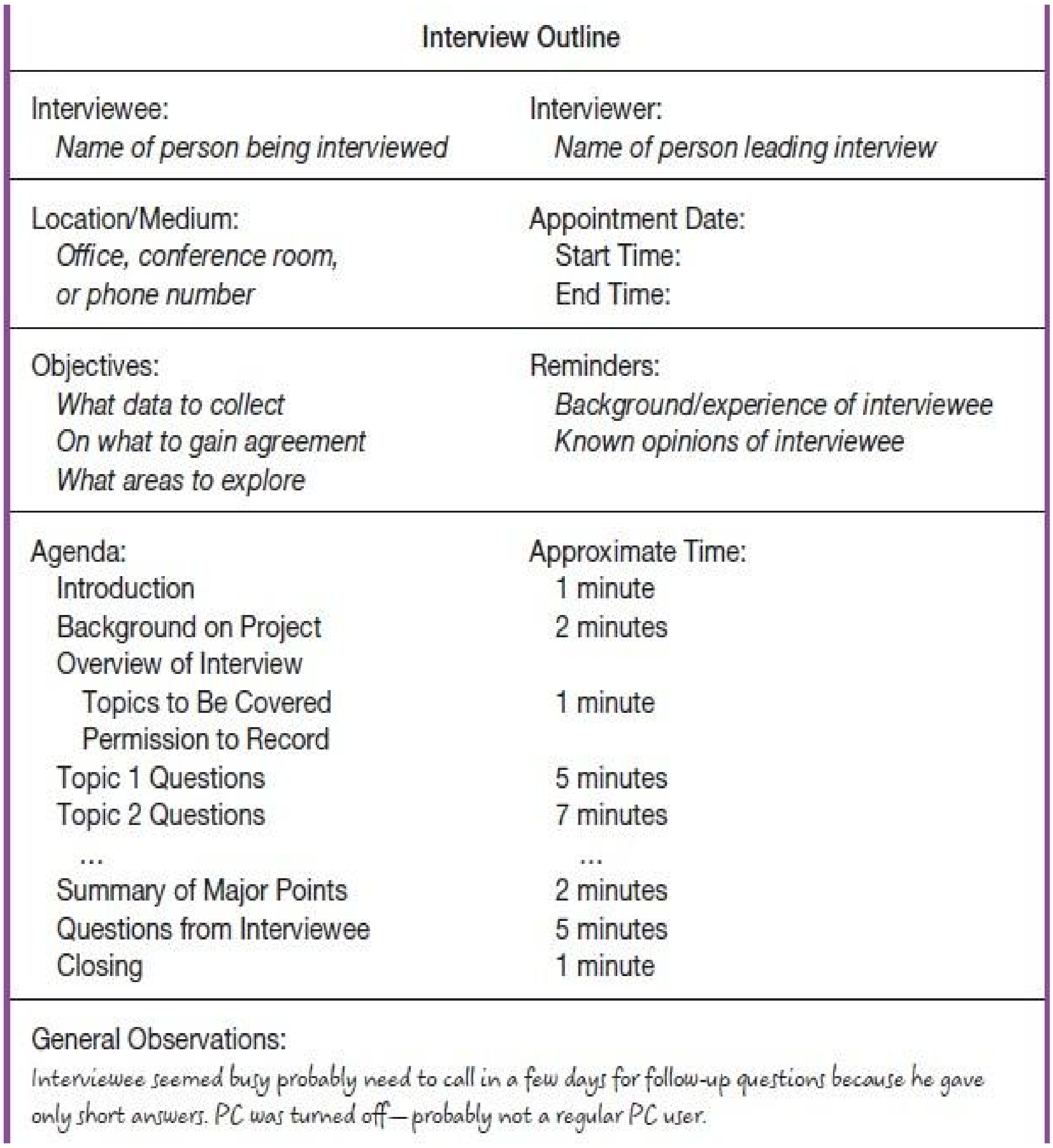
Plan the Interview
- Prepare interviewee: appointment, priming questions
- Prepare checklist, agenda, and questions

Listen carefully and take notes (record if permitted)

Review notes within 48 hours of interview

Be neutral

Seek diverse views

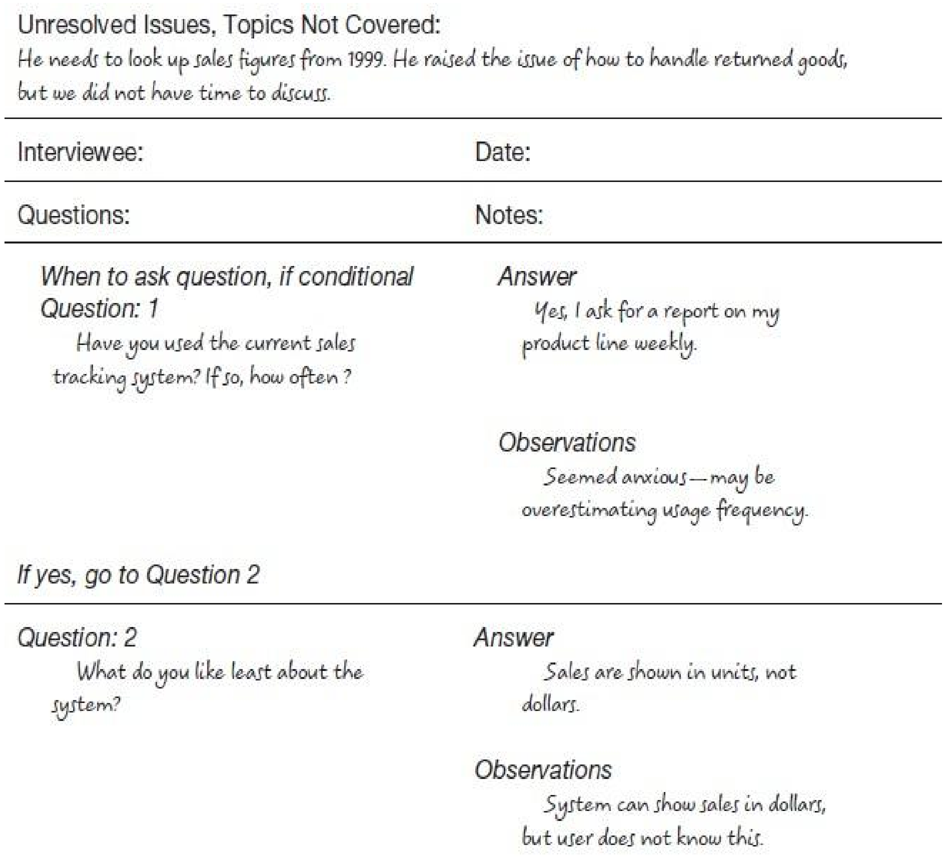


Interview Outline

Interviewee: *Name of person being interviewed*	Interviewer: *Name of person leading interview*
Location/Medium: *Office, conference room,* *or phone number*	Appointment Date: Start Time: End Time:
Objectives: *What data to collect* *On what to gain agreement* *What areas to explore*	Reminders: *Background/experience of interviewee* *Known opinions of interviewee*
Agenda:	Approximate Time:
Introduction	1 minute
Background on Project	2 minutes
Overview of Interview	
Topics to Be Covered	1 minute
Permission to Record	
Topic 1 Questions	5 minutes
Topic 2 Questions	7 minutes
...	...
Summary of Major Points	2 minutes
Questions from Interviewee	5 minutes
Closing	1 minute

General Observations:
Interviewee seemed busy probably need to call in a few days for follow-up questions because he gave only short answers. PC was turned off—probably not a regular PC user.

Figure 6-2 Typical interview guide



Unresolved Issues, Topics Not Covered:
He needs to look up sales figures from 1999. He raised the issue of how to handle returned goods, but we did not have time to discuss.

Interviewee:	Date:
Questions:	Notes:
When to ask question, if conditional *Question: 1* Have you used the current sales tracking system? If so, how often?	*Answer* Yes, I ask for a report on my product line weekly. *Observations* Seemed anxious—may be overestimating usage frequency.
If yes, go to Question 2	
Question: 2 What do you like least about the system?	*Answer* Sales are shown in units, not dollars. *Observations* System can show sales in dollars, but user does not know this.

Figure 6-2 Typical interview guide

Choosing Interview Questions

You need to decide what mix and sequence of **openended and closed-ended questions** you will use.

Open-ended questions:

Open-ended questions are usually used to **explore** for information for which you cannot anticipate all possible responses or for which you do not know the precise question to ask. The person being interviewed is encouraged to talk about whatever interests him or her within the general bounds of the question. **An example** is, "**What would you say is the best thing about the information system you currently use to do your job?**" or **"List the three most frequently used menu options."** You must react quickly to answers and determine whether or not any follow-up questions are needed for clarification or elaboration. Sometimes body language will suggest that a user has given an incomplete answer or is reluctant to disclose (to make something secret known) some information; a follow-up question might yield additional insight. One advantage of open-ended questions is that previously unknown information can surface. You can then continue exploring along unexpected lines of inquiry to reveal even more new information. **Open-ended questions also often put the interviewees at ease (move) because they are able to respond in their own words using their own structure; open-ended questions give interviewees more of a sense of involvement and control in the interview**. **A major disadvantage of open-ended questions is the length of time it can take for the questions to be answered.** In addition, open-ended questions can be difficult to summarize.

Closed-ended questions:

Closed-ended questions provide a range of answers from which the interviewee may choose. Here is an example:

Which of the following would you say is the one best thing about the information system you currently use to do **your job (pick only one)?**

a. **Having easy access to all of the data you need**

b. **The system's response time**

c. **The ability to access the system from remote locations**

Closed-ended questions work well when the major answers to questions are well known. Another plus is that interviews based on closed-ended questions **do not necessarily require a large time commitment—more topic s can be covered**. You can see body language and hear voice tone, which can aid in interpreting the interviewee's responses. Closed-ended questions can also be an easy way to begin an interview and to determine which line of open-ended questions to pursue. You can include an "other" option to encourage the interviewee to add unanticipated responses. **A major disadvantage of closed-ended questions is** that useful information that does not quite fit into the defined answers may be overlooked as the respondent tries to make a choice instead of providing his or her best answer.

Closed-ended questions, like objective questions on an examination, can follow several forms, including the following choices:

- True or false.
- Multiple choice (with only one response or selecting all relevant choices).
- Rating a response or idea on a scale, say from bad to good or strongly agree to strongly disagree. Each point on the scale should have a clear and consistent meaning to each person, and there is usually a neutral point in the middle of the scale.
- Ranking items in order of importance.

Interview Guidelines:

First, with either open- or closed-ended questions, do not express a question in a way that implies a right or wrong answer. The respondent must feel that he or she can put his or her true opinion and perspective and that his or her idea will be considered equally with those of others. Questions such as "**Should the system continue to provide the ability to override the default value**, **even though most users now do not like the feature?**".

The second guideline to remember about interviews is **to listen very carefully to what is being said**. Take careful notes or, if possible, record the interview (be sure to ask permission first!). The answers may contain extremely important information for the project.

Third, once the interview is over, **go back to your office and type up your notes within 48 hours**. If you recorded the interview, use the recording to verify the material in your notes. After 48 hours, your memory of the interview will fade quickly. As you type and organize your notes, **write down any additional questions that might arise from lapses (a temporary failure) in your notes or from ambiguous information**. Make a list of **unclear points** that need clarification. Call the person you interviewed and get answers to these new questions. Use the phone call as an opportunity to verify the accuracy of your notes. You may also want to send a written copy of your notes to the person you interviewed so the person can check your notes for accuracy. Finally, make sure you thank the person for his or her time.

Fourth, be careful during the interview not to set expectations about the new or replacement system unless you are sure these features will be part of the delivered system. **Let the interviewee know that there are many steps to the project and the perspectives of many people need to be considered, along with what is technically possible**. Let respondents know that their ideas will be carefully considered, but that due to the iterative nature of the systems development process, it is premature to say now exactly what the ultimate system will or will not do.

Fifth, seek a variety of perspectives from the interviews. Find out what potential users of the system, users of other systems that might be affected by changes, managers and superiors, information systems staff who have experience with the current system, and others think the current problems and opportunities are and what new information services might better serve the organization.

Interviewing groups:

One drawback to using interviews to collect systems requirements is the need for the analyst to reconcile (reconsider) apparent contradictions in the information collected. A series of interviews may turn up inconsistent information about the current system or its replacement. You must work through all of these inconsistencies to figure out what might be the most accurate representation of current and future systems. Such a process requires several follow-up phone calls and additional interviews. Catching important people in their offices is often difficult and frustrating, and scheduling new interviews may become very time consuming. Clearly, gathering information about an information system through a series of individual interviews and follow-up calls is not an **efficient process**.

Another option available to you is the **group interview**. In a group interview, several key people are interviewed at once. To make sure all of the important information is collected, you may conduct the interview with one or more analysts. In the case of multiple interviewers, one analyst may ask questions while another takes notes, or different analysts might concentrate on different kinds of information. For example, one analyst may listen for data requirements while another notes the timing and triggering of key events. The number of interviewees involved in the process may range from two to however many you believe can be comfortably accommodated.

A group interview has a **few advantages**.

One, it is a much more effective use of your time than a series of interviews with individuals (although the time commitment of the interviewees may be more of a concern).

Two, interviewing several people together allows them to hear the opinions of other key people and gives them the opportunity to agree or disagree with their peers. Synergies (the combined power of a group of things when they are working togetherwhich is greater than the total power achieved by each working separately) also often occur. **For example, the comments of one person might cause another person to say, "That reminds me of" or "I didn't know that was a problem."** You can benefit from such a discussion as it helps you identify issues on which there is general agreement and areas where views diverge widely.

The **primary disadvantage** of a group interview is the difficulty in scheduling it. The more people who are involved, the more difficult it will be finding a convenient time and place for everyone. Modern **videoconferencing** technology can minimize the geographical dispersion factors that make scheduling meetings so difficult. Group interviews are at the core of the JAD process, which we discuss in a later section in this chapter. A specific technique for working with groups, Nominal Group Technique, is discussed next.

Nominal Group Technique: Many different techniques have been developed over the years to improve the process of working **with groups**. One of the more popular techniques for generating ideas among group members is called **Nominal Group Technique (NGT). NGT is exactly what the name indicates—the individuals working** together to solve a problem are a group in name only, or nominally. Group members may be **gathered in the same room for NGT, but they all work alone for a period of time**. Typically, group members make a written list of their ideas. At the end of the idea-generation time, group members pool their individual ideas under the guidance of a trained facilitator. Pooling usually involves having the facilitator ask each person in turn for an idea that has not been presented before. As the person reads the idea aloud, someone else writes down the idea on a blackboard or flip chart.

After all of the ideas have been introduced, the facilitator will then ask for the group to openly discuss each. Once all of the ideas are understood by all of the participants, the facilitator will try to reduce the number of ideas the group will carry forward for additional consideration. There are many ways to reduce the number of ideas. The facilitator may ask participants to choose only a subset of ideas that they believe are important. Then the facilitator will go around the room, asking each person to read aloud an idea that is important to him or her that has not yet been identified by someone else. Or the facilitator may work with the group to identify and either eliminate or combine ideas that are very similar to others. At some point, the facilitator and the group end up with a tractable set of ideas, which can be further prioritized.

In a requirements determination context, the ideas being sought in an NGT exercise would typically apply to problems with the existing system or ideas for new features in the system being developed.

Directly observing users

For example, observation can cause people to change their normal operating behavior. Employees who know they are being observed may be nervous and make more mistakes than normal, may be careful to follow exact procedures they do not typically follow, and may work faster or slower than normal. Moreover, because observation typically cannot be continuous, you receive only a snapshot image of the person or task you observe, which may not include important events or activities.

Because observation is very time consuming, you will not only observe for a limited time, but also a limited number of people and a limited number of sites. Again, observation yields only a small segment of data from a possibly vast variety of data sources. Exactly which people or sites to observe is a difficult selection problem. You want to pick both typical and atypical people and sites,and observe during normal and abnormal conditions and times to receive the richest possible data from observation.

Analyzing Procedures and other Documents

- By examining existing system and organizational documentation, system analyst can find out details about current system and the organization these systems supports. In documents analyst can find information such as problem with existing systems, opportunities to meet new needs if only certain information or information processing were available, organizational direction that can influence information system requirements, and the reason why current systems are designed as they are, etc.
- However, when analyzing those official documentations, analysts should pay attention to the difference between the systems described on the official documentations and the practical systems in real world. For the reason of inadequacies (not enough) of formal procedures, individual work habits and preferences, resistance to control, and other factors, the difference between so

called formal system and informal system universally exists.

- In documents you can find information about problems with existing systems (e.g., missing information or redundant steps)
- **Opportunities to meet new needs if only certain information or information processing were available** (e.g., analysis of sales based on customer type)
- Organizational direction that can influence information system requirements (e.g., trying to link customers and suppliers more closely to the organization)
- Titles and names of key individuals who have an interest in relevant existing systems (e.g., the name of a sales manager who led a study of the buying behavior of key customers)
- Values of the organization or individuals who can help determine priorities for different capabilities desired by different users (e.g., maintaining market share even if it means lower short-term profits)
- Special information processing circumstances that occur irregularly that may not be identified by any other requirements determination technique (e.g., special handling needed for a few large-volume customers that requires use of customized customer ordering procedures)
- The reason why current systems are designed as they are, which can suggest features left out of current software, which may now be feasible and more desirable (e.g., data about a customer's purchase of competitors' products were not available when the current system was designed; these data are now available from several sources)
- Data, rules for processing data, and principles by which the organization operates that must be enforced by the information system (e.g., each customer is assigned exactly one sales department staff member as a primary contact if the customer has any questions)

Contemporary Methods For Determining system Requirements

Even though we called interviews, observation, and document analysis traditional methods for determining a system's requirements, all of these methods are still very much used by analysts to collect important information. Today, however, there are additional techniques to collect information about the current system, the organizational area requestingthe new system, and what the new system should be like.

In this section, you will learn about several contemporary information- gathering techniques for analysis (listed in Table 6-5): **JAD**, **CASE tools** to support JAD, and **prototyping**.

As we said earlier, these techniques can support effective information collection and structuring while reducing the amount of time required foranalysis.

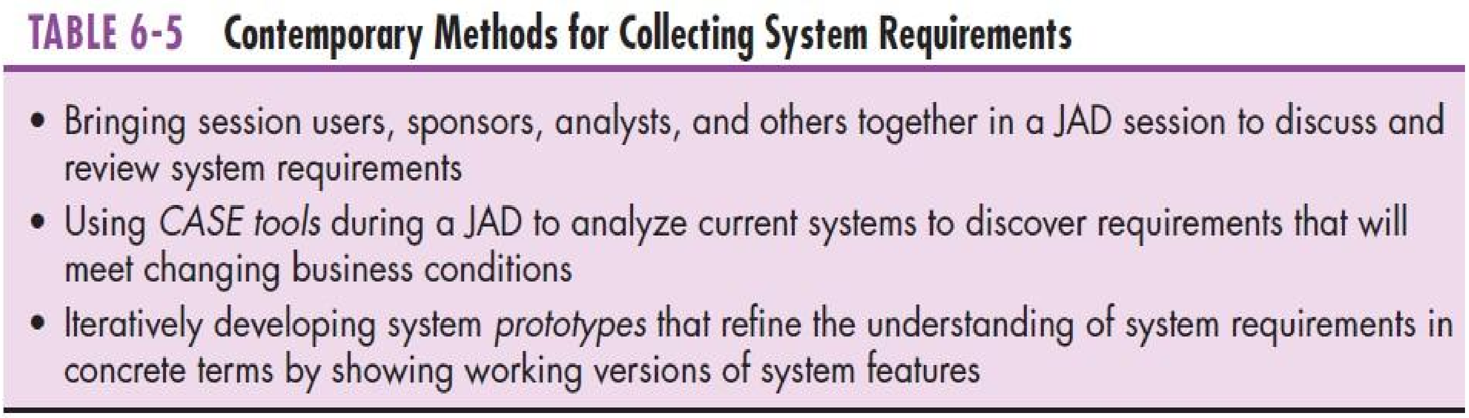


TABLE 6-5 Contemporary Methods for Collecting System Requirements

- Bringing session users, sponsors, analysts, and others together in a JAD session to discuss and review system requirements
- Using *CASE tools* during a JAD to analyze current systems to discover requirements that will meet changing business conditions
- Iteratively developing system *prototypes* that refine the understanding of system requirements in concrete terms by showing working versions of system features

Joint Application Design (JAD)

A structured process in which users, managers, and analysts work together for several days in a series of intensive meetings to specify or review system requirements.

Joint Application Design (**JAD**) started in the late 1970s at IBM, and since then the practice of JAD has spread throughout many companies and industries.

The main idea behind JAD is to bring together the **key users**, **managers**, and **systems analysts** involved in the analysis of a current system. The primary purpose of using JAD in the analysis phase is to collect systems requirements simultaneously from the key people involved with the system. The result is an intense (extreme and forceful or (of a feeling) very strong) **and structured, but highly effective, process**. As with a group interview, having all the key people together in one place at one time allows analysts to see where there are areas of agreement and where there are conflicts. Meeting with all of these important people forover a week of intense **sessions** allows you the opportunity to resolve conflicts, or at least to understand why a conflict may not be simple to resolve.

JAD Sessions:

JAD sessions are usually conducted at a location other than the place where the people involved normally work. The idea behind such a practice is **to keep participants away from as many distractions** as possible so that they can concentrate on systems analysis. A JAD may last anywhere from **four hours to an entire week** and may consist of several sessions. **A JAD employs thousands of dollars of corporate resources**, the most expensive of which is the time of the people involved. Other expenses include the costs associated with flying people to a remote site and putting them up in hotels and feeding them for several days. The **typical participants in a JAD** are listed below:

JAD session leader: The JAD session leader organizes and runs the JAD. This person has been trained in group management and facilitation as well as in systems analysis. The JAD leader sets the agenda and sees that it is met; he or she remains neutral on issues and does not contribute ideas or opinions, but rather concentrates on keeping the group on the agenda, resolving conflicts and disagreements, and soliciting (manage) all ideas.

Users: The key users of the system under consideration are vital participants in a JAD. They are the only ones who have a clear understanding of what it means to use the system on a daily basis.

Managers: Managers of the work groups who use the system in questionprovide insight into new organizational directions, motivations for and organizational impacts of systems, and support for requirements determined during the JAD.

Sponsor: As a major undertaking due to its expense, a JAD must be sponsored by someone at a relatively high level in the company. If the sponsor attends any sessions, it is usually only at the very beginning or the end.

Systems analysts: Members of the systems analysis team attend the JAD,although their actual participation may be limited. Analysts are there to learn from users and managers, not to run or dominate the process.

Scribe: The scribe **takes notes during the JAD sessions**. This is usually done on a laptop. Notes may be taken using a word processor, or notes and diagrams may be entered directly into a **CASE tool**.

IS staff: Besides systems analysts, other **information systems** (IS) staff, such as **programmers**, **database analysts**, **IS planners**, and **data center personnel,** may attend to learn from the discussion and possibly contribute their ideas on the technical feasibility of proposed ideas or the technical limitations of current systems.

JAD sessions are usually held in special-purpose rooms where participants sit around **horseshoe-shaped** tables, as shown in **Figure 6-6**. These rooms are typically equipped with whiteboards. Other audiovisual tools may be used, such as **magnetic symbols** that can be easily **rearranged on a whiteboard**, flip charts, and computer-generated displays. **Flip-chart paper is typically used for keeping track of issues that cannot be resolved** during the JAD or for those issues requiring additional information that can be gathered during breaks in the proceedings. Computers may be used to create and display form or report designs, diagram existing or replacement systems,or create prototypes.

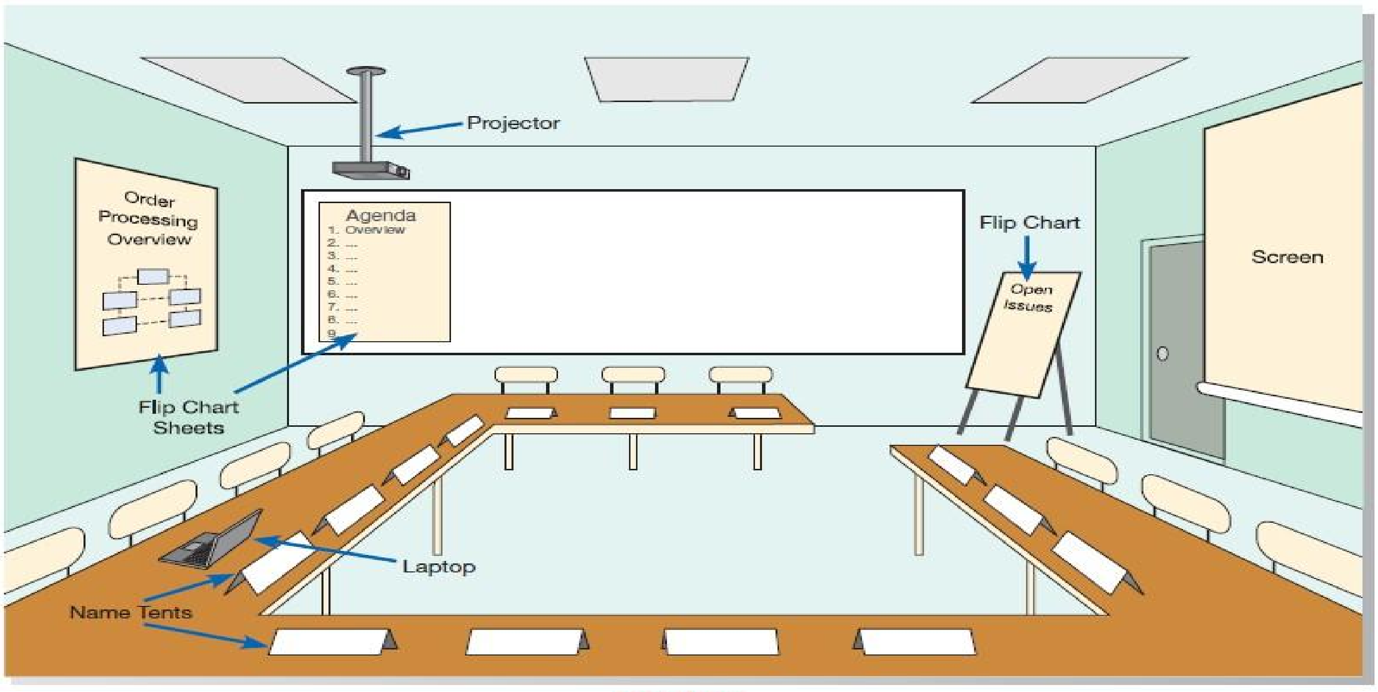


FIGURE 6-6
Illustration of the typical room layout for a JAD
(*Source:* Based on Wood and Silver, 1995.)

When a JAD is completed, the end result is a set of documents that detail the workings of the current system related to the study of a replacement system. Depending on the exact purpose of the JAD, analysts may also walk away from the JAD with some detailed information on what is desired of the replacement system.

CASE Tools During JAD: For requirements determination and structuring, the most useful CASE tools are used for diagramming and form and report generation.

Some observers advocate using CASE tools during JADs (Lucas, 1993). Running a CASE tool during a JAD enables analysts to enter system models directly into a CASE tool, providing consistency and reliability in the joint model-building process.

The CASE tool captures system requirements in a **more flexible** and **useful** way than can a scribe or an analysis team **making notes**. Further, the CASE tool can be used to project **menu**, **display**, and **report designs**, so users can directly observe **old** and new **designs** and evaluate their usefulness for the analysis team.

Advantage of JAD

- JAD allows you to resolve difficulties more simply and produce better, error-free software.
- The joint collaboration between the company and the clients lowers all risks.
- JAD reduces costs and time needed for project development.
- Well-defined requirements improve system quality.
- Due to the close communication, progress is faster.
- JAD encourages the team to push each other to work faster and deliver on time.

Disadvantage of JAD

- Different opinions within the team make it difficult to align goals and maintain focus
- Depending on the size of the project , JAD may require a significant time commitment.

Using Prototyping During Requirements Determination:

Prototyping: It is an **iterative process of systems development in which requirements are converted to a working system that is continually revised through close collaboration between an analyst and users**. Prototyping will enable you to quickly convert basic requirements into a working, though limited, version of the desired information system. **The prototype will then be viewed and tested by the user**. Typically, seeing verbal descriptions of requirements converted into a physical system will prompt the user to modify existing requirements and generate new ones. For example, in the initial interviews, a user might have said that he wanted all relevant utility billing information (e.g., the client's name and address, the service record, and payment history) on a single computer display form. Once the same user sees how crowded and confusing such a design would be in the prototype, he might change his mind andinstead ask to have the information organized on several screens, but with easy transitions from one screen to another. He might also be reminded of some important requirements (data, calculations, etc.) that had not surfaced during the initial interviews.

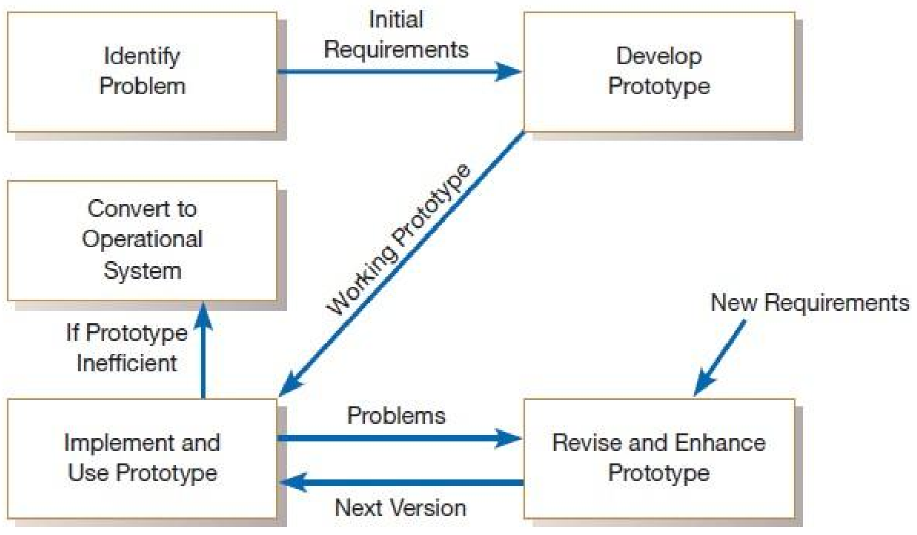


FIGURE 6-7
The prototyping methodology
(*Source:* Based on "Prototyping: The New Paradigm for Systems Development," by J. D. Naumann and A. M. Jenkins, *MIS Quarterly* 6(3): 29–44.)

As the prototype **changes through each iteration, more and more of the design specifications for the system are captured in the prototype**. The prototype can then serve as the basis for the production system in a process called **evolutionary prototyping**.

Alternatively, the prototype can serve only as a model, which is then used as a reference for the construction of the actual system. In this process, **called throwaway prototyping**, the prototype is discarded after it has been used.

Evolutionary Prototyping:

In **evolutionary prototyping, you begin by modeling parts of the target system and, if the prototyping process is successful, you evolve the rest of the system from those parts** (McConnell, 1996). A life-cycle model of evolutionary prototyping illustrates the iterative nature of the process and the tendency to refine the prototype until it is ready to release (Figure 6-8).Systems must be designed to support scalability, multiuser support, and multiplatform support. Few of these design specifications will be coded into a prototype. Further, as much as 90 percent of a system's functioning is devoted to handling exceptional cases (McConnell, 1996). **Prototypes are designed to handle only the typical cases, so exception handling must be added to the prototype as it is converted to the production system. Clearly, the prototype captures only part of the system requirements.**

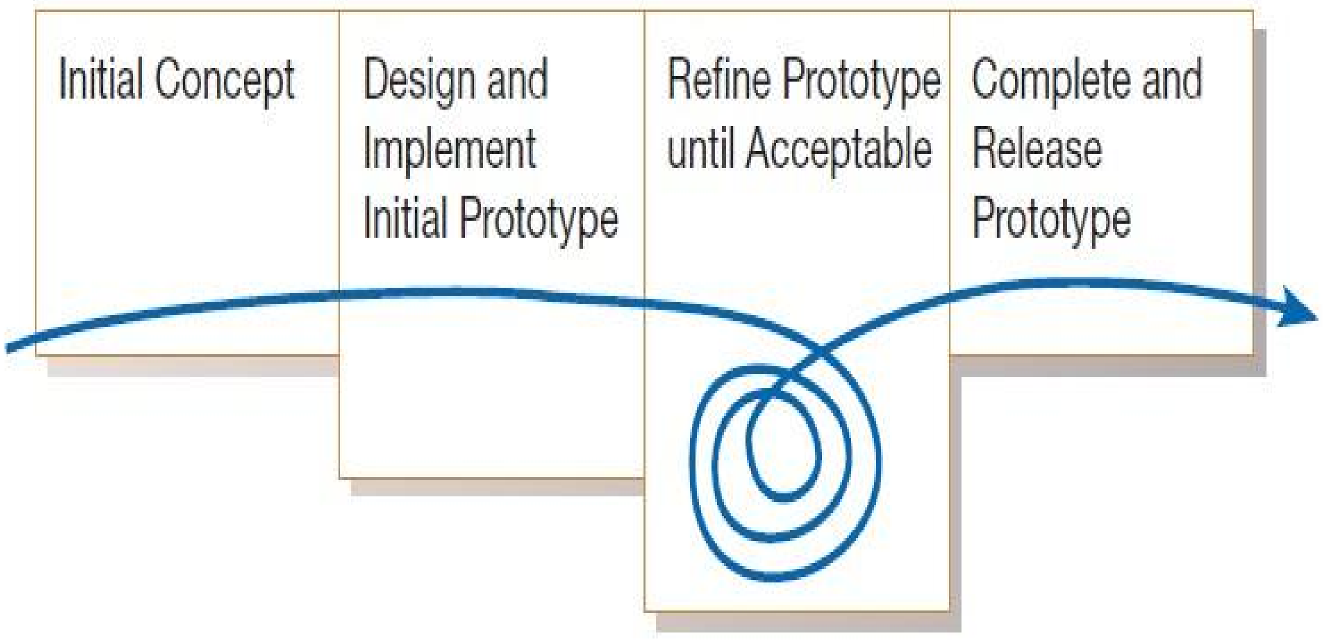

Initial Concept
Design and Implement Initial Prototype
Refine Prototype until Acceptable
Complete and Release Prototype

FIGURE 6-8
McConnell's evolutionary prototyping model

Throwaway Prototyping :

Unlike evolutionary prototyping, throwaway prototyping does not preserve the prototype that has been developed. With throwaway prototyping, there is never any intention to convert the prototype into a working system.

Instead**, the prototype is developed quickly to demonstrate some aspect of a system design that is unclear or to help users decide among differentfeatures or interface characteristics**. Once the uncertainty the prototype was created to address has been reduced, the prototype can be discarded, and the principles learned from its creation and testing can then become part of the requirements determination.

Radical Methods For Determining System Requirements

The overall **process by which current methods are replaced with radicallynew methods is generally referred to as business process reengineering (BPR).**To better understand BPR, consider the following analogy.

Suppose you are a successful European golfer who has tuned your game to fit the style of golf courses and weather in Europe. You have learned how to control the flight of the ball in heavy winds, roll the ball on wide open greens, putt on large and undulating greens, and aim at a target without the aid of the landscaping common on North American courses. When you cometo the United States to make your fortune on the US tour, you discover that incrementally improving your putting, driving accuracy, and sand shots will help, but the new competitive environment is simply not suited to your style of the game. You need to reengineer your whole approach, learning how to aim at targets, spin and stop a ball on the green, and manage the distractions of crowds and the press. If you are good enough,you may survive, but without reengineering, you will never be a winner.

Just as the competitiveness of golf forces good players to adapt their games to changing conditions, the competitiveness of our global economy has driven most companies into a mode of continuously improving the quality of their products and services (Dobyns and Crawford-Mason, 1991). Organizations realize that creatively using information technologies can yield significant improvements in most business processes.

Identifying Processes To Reengineer

A first step in any BPR (business process reengineering) effort **relates to understanding what processes to change**. To do this, you must first understand which processes represent the key business processes for the organization. **Key business processes are the structured set of measurable** activities designed to produce a specific output for a particular customer or market. **The important aspect of this definition is that key processes are focused on some type of organizational outcome, such as the creation of a product or the delivery of a service**. Key business processes are also customer focused. **In other words**, key business processes would include all activities used to **design**, **build**, **deliver**, **support**, and **service** a particular product for a particular customer.

Disruptive technologies

Once key business processes and activities have been identified, information technologies must be applied to radically improve business processes. To do this, Hammer and Champy (1993) suggest that organizations think "inductively" about information technology. **Induction is the process of reasoning from the specific to the general, which means that managers must learn the power of new technologies and think of innovative (using new methods or ideas) ways to alter the way work is done**. **This is contrary to deductive thinking, where problemsare first identified and solutions are then formulated.**

Requirements Management Tools

Organizations and developers have always been looking for more effective and creative ways to create and maintain requirements documents. One method that has been developed is computer-based requirements management tools. These tools make it easier for analyst to keep requirements, current documents and to define links between different parts of the overall specifications package. Requirements management tools are typically designed to work with many of the standards now available for requirements specification such as the **Unified Modeling Language 2.0, System modeling language, Business process modeling notation etc.**

Requirements management tools is best to traditional, planning based systems development approaches. They do not work well with **agile methodologies** which tend to employ different approaches to requirements gathering.

Requirements Determination Using Agile Methodologies

Three techniques are presented in this section.

The first is continual user involvement in the development process, a technique that works especially well with **small and dedicated development teams**.

The second approach is a JAD-like process called Agile Usage-Centered Design.

The third approach is the Planning Game, which was developed as part of eXtreme Programming.

Continual User Involvement

As the **criticisms** of the traditional waterfall SDLC. One of those criticisms was that the waterfall **SDLC allowed users to be involved in the development process only in the early stages of analysis**. Once requirements had been gathered from them, the users were not involved again in the process until the system was being installed and they were asked to sign off on it. Typically, by the time the users saw the system again, it was nothing like what they had imagined. The system most likely did not adequately address user needs. One approach to the problem of limited user involvement is to involve the users continually, throughout the entire analysis and design process. Such an approach works best when development can follow the **analysis–design–code–test cycle** favored by the Agile Methodologies (**Figure 6-9**), because the user can provide information on requirements and then watch and evaluate as those requirements are designed, coded,and tested. This iterative process can continue through several cycles, until most of the major functionality of the system has been developed. Extensive involvement users in the analysis and design process is a key part of many **Agile approaches**, but it was also a key part of **Rapid Application Development** (see Chapter 1).

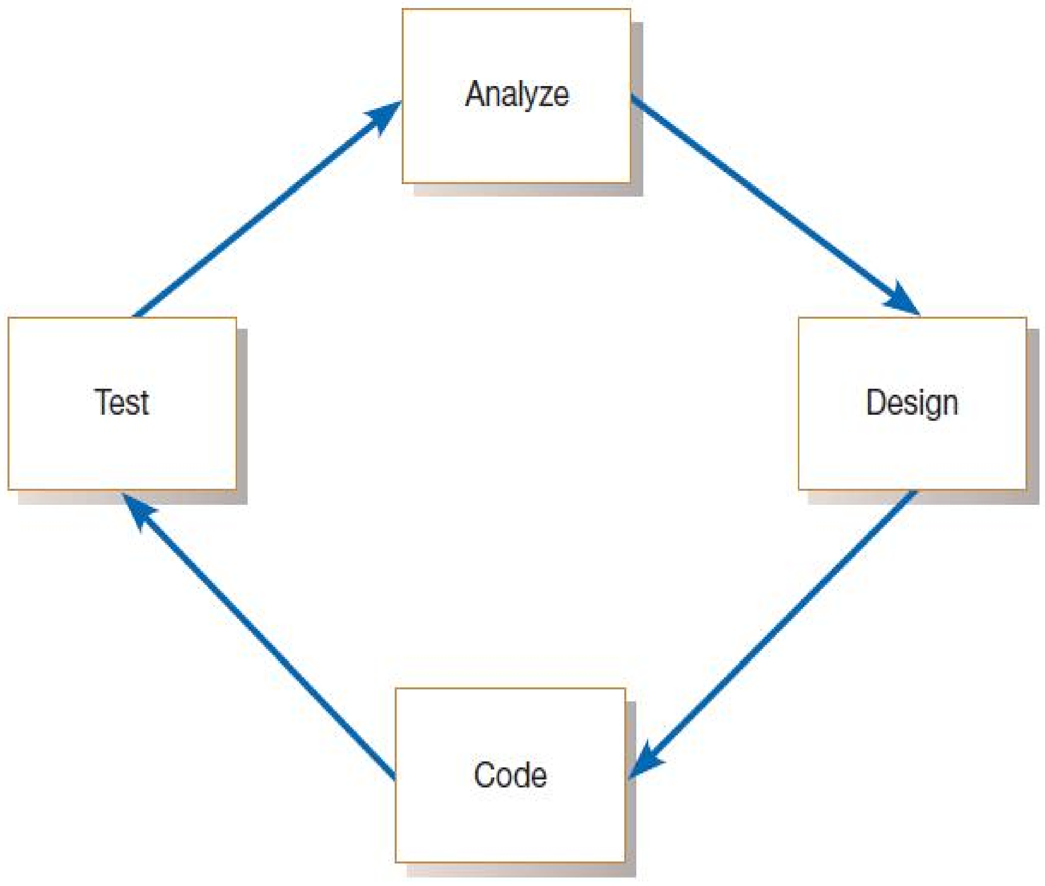


FIGURE 6-9
The iterative analysis–design–code–test cycle

Continual user involvement was a key aspect of the success of Boeing's Wire Design and Wire Install system for the 757 aircraft (Bedoll, 2003). The system was intended to support engineers who customize plane configurations for customers, allowing them to analyze all 50,000 wires that can possibly be installed on a 757. A previous attempt at building a similar system took **over three years**, and the resulting system **was never used**. The second attempt, relying on Agile Methodologies, resulted in a system that was in production after only **six weeks**. One of the keys to success was a user liaison (communication between people who work with each other) who spent **half of his time with the small development team and half with the other end users**. In addition to following the analysis–design–code–test cycle, the team also **had weekly production releases**. The user liaison was involved every step of the way. Obviously, for such a requirements determination to succeed, the user who works with the development **team must be very knowledgeable**, but he or she must also be in a position to give up his or her normal business responsibilities in order to become heavily involved in the system's development.

Agile Usage-Centered Design

Continual user involvement in systems development is an excellent way to ensure that **requirements are captured accurately and immediately implemented in system design**. However, such constant interaction works best when the **development team is small**, as was the case in the Boeing example. Also, it is not always possible to have continual access tousers for the duration of a development project. Thus, **Agile developers have come up with other means for effectively involving users in the requirements determination process**. One such method is called **Agile Usage-Centered Design**, originally developed by Larry Constantine (2002)and adapted for Agile Methodologies by Jeff Patton (2002).

The Planning Game From eXtreme Programming

You read about eXtreme Programming in Chapter 1, and you know that it is an approach to software development put together by Kent Beck (Beck and Andres, 2004). You also know that it is distinguished by **its short cycles, its incremental planning approach**, **its focus on automated testswritten by programmers** and **customers to monitor the process of development**, and its reliance on an **evolutionary approach** to development that lasts throughout the lifetime of the system. One of the key emphases of **eXtreme Programming** is its use of **two-person programming teams** and having a customer on-site during the development process. The relevant parts of eXtreme Programming that relate to requirements determination are

(1) **how planning, analysis, design, and construction are all fused together into a single phase of activity** **a**nd

(2**) its unique way of capturing and presenting system requirements and design specifications**. All phases of the life cycle converge into a series of activities based on the basic processes of coding, testing, listening, and designing.