Introduction to Systems Analysis and Design

Introduces the systems development life cycle (SDLC), the fundamental four-phase model (planning, analysis, design, and implementation) common to all information system development projects. Secondly, it describes the evolution of system development methodologies. Thirdly, overviews object-oriented systems analysis and design and describes the Unified Process and its extensions.

The systems development life cycle (SDLC)

Is the process of understanding how an information system (IS) can support business needs by designing a system, building it, and delivering it to users. If you have taken a programming class or have programmed on your own, this probably sounds pretty simple. Unfortunately, it is not. A 1996 survey by the Standish Group found that 42 percent of all corporate IS projects were abandoned before completion. A similar study done in 1996 by the General Accounting Office found 53 percent of all U.S. government IS projects were abandoned. Unfortunately, many of the systems that aren't abandoned are delivered to the users significantly late, cost far more than planned, and have fewer features than originally planned.

The key person in the SDLC is the systems analyst, who analyzes the business situation, identifies opportunities for improvements, and designs an information system to implement them. Being a systems analyst is one of the most interesting, exciting, and challenging jobs around. Systems analysts work with a variety of people and learn how they conduct business. Specifically, they work with a team of systems analysts, programmers, and others on a common mission. Systems analysts feel the satisfaction of seeing systems that they designed and developed

make a significant business impact, knowing that they contributed unique skills to make that happen.

However, the primary objective of a systems analyst is not to create a wonderful sys- tem; instead, it is to create value for the organization, which for most companies means increasing profits (government agencies and not-for-profit organizations measure value differently). Many failed systems have been abandoned because the analysts tried to build a wonderful system without clearly understanding how the system would fit with an organization's goals, current business processes, and other information systems to provide value. An investment in an information system is like any other investment, such as a new machine tool. The goal is not to acquire the tool, because the tool is simply a means to an end; the goal is to enable the organization to perform work better so it can earn greater profits or serve its constituents more effectively.

THE SYSTEMS DEVELOPMENT LIFE CYCLE

In many ways, building an information system is similar to building a house. First, the house (or the information system) starts with a basic idea. Second, this idea is transformed into a simple drawing that is shown to the customer and refined (often through several drawings, each improving on the last) until the customer agrees that the picture depicts what he or she wants. Third, a set of blueprints is designed that presents much more detailed information about the house (e.g., the type of water faucets, where the telephone jacks will be placed). Finally, the house is built following the blueprints, often with some changes directed by the customer as the house is erected.

The SDLC has a similar set of four fundamental *phases*: planning, analysis, design, and implementation. Different projects may emphasize different parts of the SDLC or approach the SDLC phases in different ways, but all projects have elements of these four phases. Each *phase* is itself composed of a series of *steps*,

which rely upon *techniques* that produce deliverables *(*specific documents and files that provide understanding about the project).

For example, when you apply for admission to a university, all students go through the same phases: information gathering, applying, and accepting. Each of these phases has steps information gathering includes steps such as searching for schools, requesting information, and reading brochures. Students then use techniques (e.g., Internet searching) that can be applied to steps (e.g., requesting information) to create *deliverables* (e.g., evaluations of different aspects of universities).

Planning

The *planning phase* is the fundamental process of understanding *why* an information system should be built and determining how the project team will go about building it. It has two steps:

1. During ***project initiation***, the system's business value to the organization is identified: how will it lower costs or increase revenues? Most ideas for new systems come from outside the IS area (from the marketing department, accounting department, etc.) in the form of a *system request.* A system request presents a brief summary of a business need, and it explains how a system that supports the need will create business value. The IS department works together with the person or department that generated the request (called the *project sponsor*) to conduct a *feasibility analysis.* The feasibility analysis examines key aspects of the proposed project:

- The idea's technical feasibility (Can we build it?)
- The economic feasibility (Will it provide business value?)
- The organizational feasibility (If we build it, will it be used?)

The system request and feasibility analysis are presented to an information

systems *approval committee* (sometimes called a steering committee), which decides whether the project should be undertaken.

2. Once the project is approved, it enters ***project management***. During project management, the *project manager* creates a *work plan*, staffs the project, and puts techniques in place to help the project team control and direct the project through the entire SDLC. The deliverable for project management is a *project plan*, which describes how the project team will go about developing the system.

Analysis

The *analysis phase* answers the questions of *who* will use the system, *what* the system will do, and *where* and *when* it will be used. During this phase, the project team investigates any current system(s), identifies improvement opportunities, and develops a concept for the new system

This phase has three steps

1. An ***analysis strategy*** is developed to guide the project team's efforts. Such a strategy usually includes an analysis of the current system (called the *as-is system*) and its problems, and then ways to design a new system (called the *to-be system*).
2. The next step is ***requirements gathering*** (e.g., through interviews or questionnaires). The analysis of this information—in conjunction with input from project sponsor and many other people—leads to the development of a concept for a new system. The system concept is then used as a basis to develop a set of business *analysis models*, which describe how the business will operate if the new system is developed. The set of models typically includes models that represent the data and processes necessary to support the underlying business process.
3. The analyses, system concept, and models are combined into a document called the ***system proposal,*** which is presented to the project sponsor and other key decision makers (e.g., members of the approval committee) who

decide whether the project should continue to move forward.

The system proposal is the initial deliverable that describes what business requirements the new system should meet. Because it is really the first step in the design of the new system, some experts argue that it is inappropriate to use the term analysis as the name for this phase; some argue a better name would be analysis and initial design. Most organizations continue use to the name *analysis* for this phase, however, so we use it in this book as well. Just keep in mind that the deliverable from the analysis phase is both an analysis and a high-level initial design for the new system.

Design

The *design phase* decides *how* the system will operate, in terms of the hardware, software, and network infrastructure; the user interface, forms and reports; and the specific programs, data- bases, and files that will be needed. Although most of the strategic decisions about the system were made in the development of the system concept during the analysis phase, the steps in the design phase determine exactly how the system will operate. The design phase has four steps:

1. The ***design strategy*** is first developed. It clarifies whether the system will be developed by the company's own programmers, whether the system will be outsourced to another firm (usually a consulting firm), or whether the company will buy an existing software package.
2. This leads to the development of the basic ***architecture design*** for the system, which describes the hardware, software, and network infrastructure to be used. In most cases, the system will add or change the infrastructure that already exists in the organization. The *interface design* specifies how the users will move through the system (e.g., navigation methods such as menus and on-screen buttons) and the forms and reports that the system will use.
3. **The *database and file specifications*** are developed. These define exactly

what data will be stored and where they will be stored.

4. The analyst team develops the ***program design***, which defines the programs that need to be written and exactly what each program will do.

This collection of deliverables (architecture design, interface design, database and file specifications, and program design) is the ***system specification*** that is handed to the programming team for implementation. At the end of the design phase, the feasibility analysis and project plan are reexamined and revised, and another decision is made by the project sponsor and approval committee about whether to terminate the project or continue.

Implementation

The final phase in the SDLC is the ***implementation phase***, during which the system is actually built (or purchased, in the case of a packaged software design). This is the phase that usually gets the most attention, because for most systems it is the longest and most expensive single part of the development process. This phase has three steps:

1. System ***construction*** is the first step. The system is built and tested to ensure it per- forms as designed. Because the cost of bugs can be immense, testing is one of the most critical steps in implementation. Most organizations give more time and attention to testing than to writing the programs in the first place.
2. The system is installed. ***Installation*** is the process by which the old system is turned off and the new one is turned on. It may include a direct cutover approach (in which the new system immediately replaces the old system), a parallel conversion approach (in which both the old and new systems are operated for a month or two until it is clear that there are no bugs in the new system), or a phased conversion strategy (in which the new system is installed in one part of the organization as an initial trial and then gradually

installed in others). One of the most important aspects of conversion is the development of a *training plan* to teach users how to use the new system and help manage the changes caused by the new system.

3. The analyst team establishes a ***support plan*** for the system. This plan usually includes a formal or informal post-implementation review as well as a systematic way for identifying major and minor changes needed for the system.

System Development Methodology

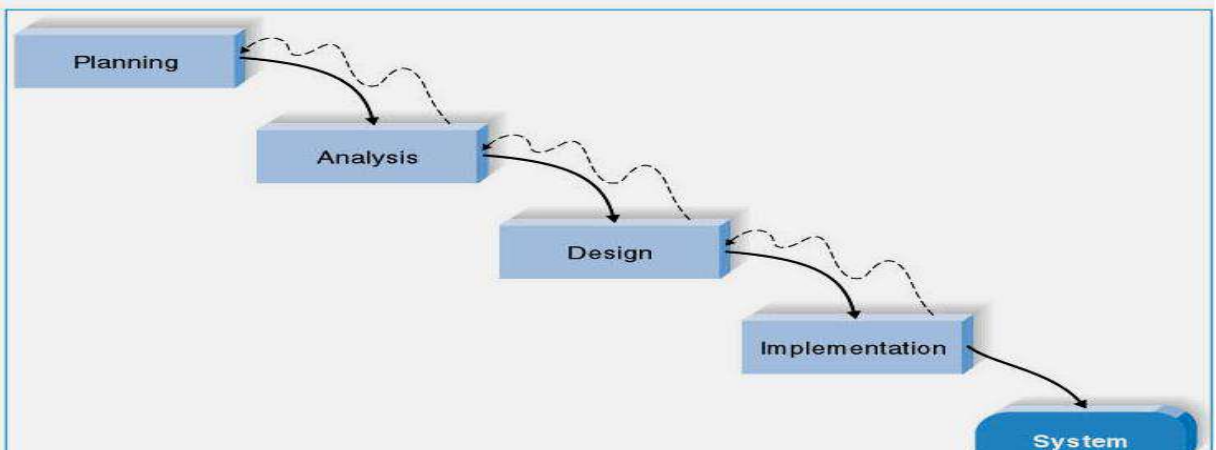
A *methodology* is a formalized approach to implementing the SDLC (i.e., it is a list of steps and deliverables). There are many different systems development methodologies, and each one is unique, based on the order and focus it places on each SDLC phase. Some methodologies are formal standards used by government agencies, whereas others have been developed by consulting firms to sell to clients. Many organizations have internal methodologies that have been honed over the years, and they explain exactly how each phase of the SDLC is to be performed in that company.

There are many ways to categorize methodologies. One way is by looking at whether they focus on business processes or the data that support the business. A ***process-centered methodology*** emphasizes process models as the core of the system concept. For example, process-centered methodologies would focus first on defining the processes (e.g., assemble sandwich ingredients). *Data-centered* methodologies emphasize data models as the core of the system concept. In data centered methodologies would focus first on defining the contents of the storage areas (e.g., refrigerator) and how the contents were organized. By contrast, ***object-oriented methodologies*** attempt to balance the focus between process and data by incorporating both into one model.

Structured Design

The first category of systems development methodologies is called *structured design.* These methodologies became dominant in the 1980s, replacing the previous, ad-hoc, and undisciplined approach. Structured design methodologies adopt a formal step-by-step approach to the SDLC that moves logically from one phase to the next. Numerous process-centered and data-centered methodologies follow the basic approach of the two structured design categories outlined next.

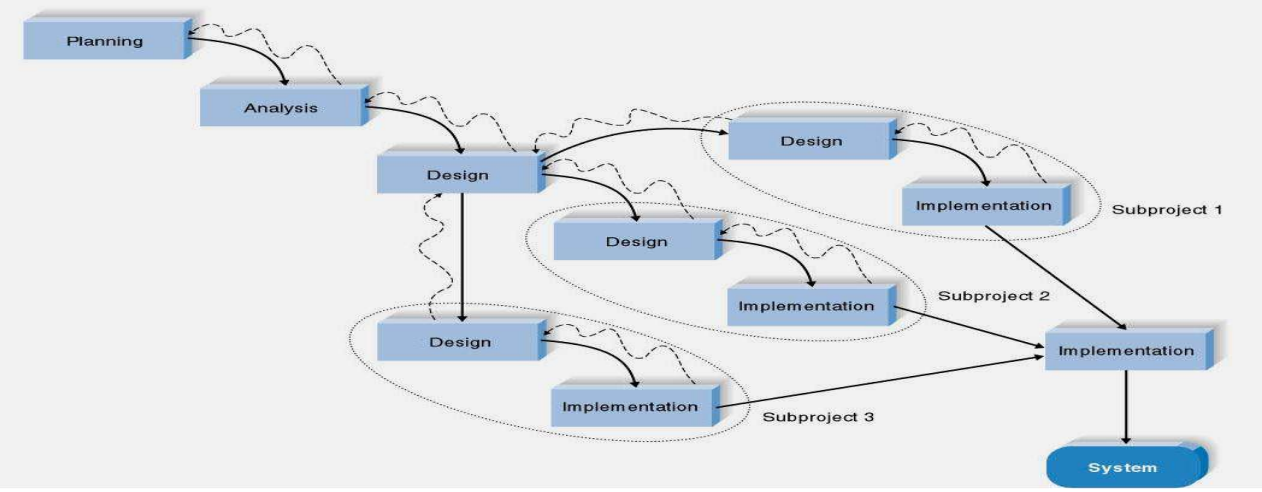
1. **Waterfall Development** The original structured design methodology (still used today) is *waterfall development.* With waterfall development–based methodologies, the analysts and users proceed in sequence from one phase to the next. The key deliverables for each phase are typically very long (often hundreds of pages in length) and are presented to the project sponsor for approval as the project moves from phase to phase. Once the sponsor approves the work that was conducted for a phase, the phase ends and the next one begins. This methodology is referred to as waterfall development because it moves forward from phase to phase in the same manner as a waterfall. Although it is possible to go backward in the SDLC (e.g., from design back to analysis), it is extremely difficult (imagine yourself as a salmon trying to swim upstream against a waterfall, as shown in Figure below.



The two key advantages of the structured design waterfall approach are that it an identifies system requirement long before programming begins and it minimizes changes to the requirements as the project proceeds. The two key disadvantages are that the design must be completely specified before programming begins and that a long time elapses between the completion of the system proposal in the analysis phase and the delivery of the system (usually many months or years).

2. **Parallel Development:** *Parallel development* methodology attempts to address the problem of long delays between the analysis phase and the delivery of the system. Instead of doing design and implementation in sequence, it performs a general design for the whole system and then divides the project into a series of distinct subprojects that can be designed and implemented in parallel. Once all subprojects are complete, there is a final integration of the separate pieces, and the system is delivered shown below.

   The primary advantage of this methodology is that it can reduce the schedule time to deliver a system; thus, there is less chance of changes in the business environment causing rework. However, the approach still suffers from problems caused by paper documents. It also adds a new problem: Sometimes the subprojects are not completely independent; design decisions made in one subproject may affect another, and the end of the project may require significant integration efforts.

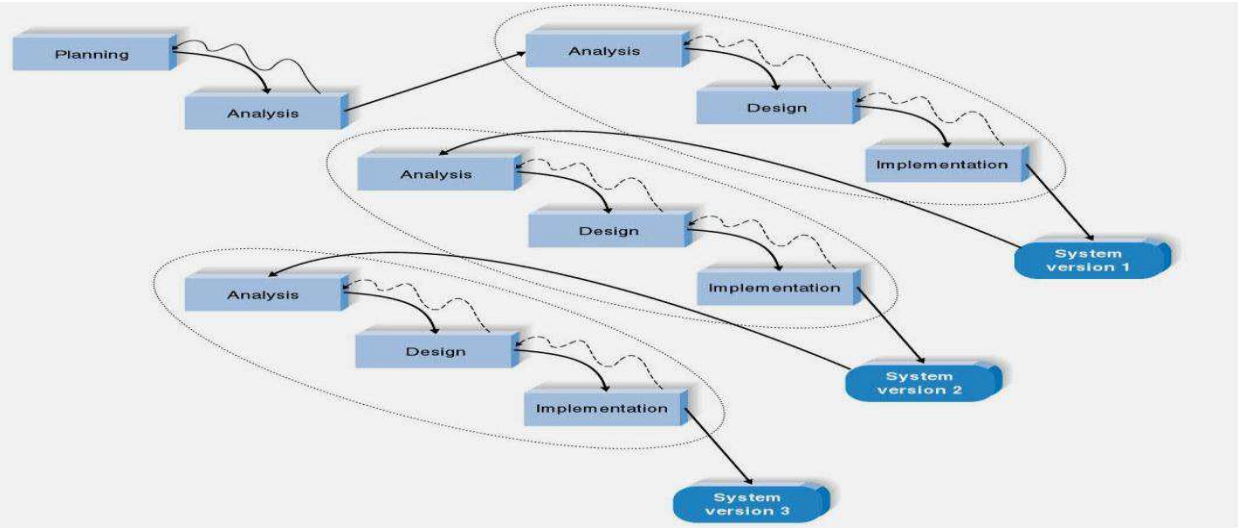


A Parallel Development–based Methodology

Rapid Application Development (RAD)

A second category of methodologies includes *rapid application development (RAD)*–based methodologies. These are a newer class of systems development methodologies that emerged in the 1990s. RAD-based methodologies attempt to address both weaknesses of structured design methodologies by adjusting the SDLC phases to get some part of the system developed quickly and into the hands of the users. In this way, the users can better understand the system and suggest revisions that bring the system closer to what is needed.

Most RAD-based methodologies recommend that analysts use special techniques and computer tools to speed up the analysis, design, and implementation phases, such as CASE tools, joint application design (JAD) sessions, fourth-generation/visual programming languages that simplify and speed up programming (e.g., Visual Basic), and code generators that automatically produce programs from design specifications. The combination of the changed SDLC phases and the use of these tools and techniques improve the speed and quality of systems development. However, there is one possible subtle problem with RAD-based methodologies: managing user expectations. Due to the use of the tools and techniques that can improve the speed and quality of systems development, user expectations of what is possible may dramatically change.



A Phased Development–based Methodology

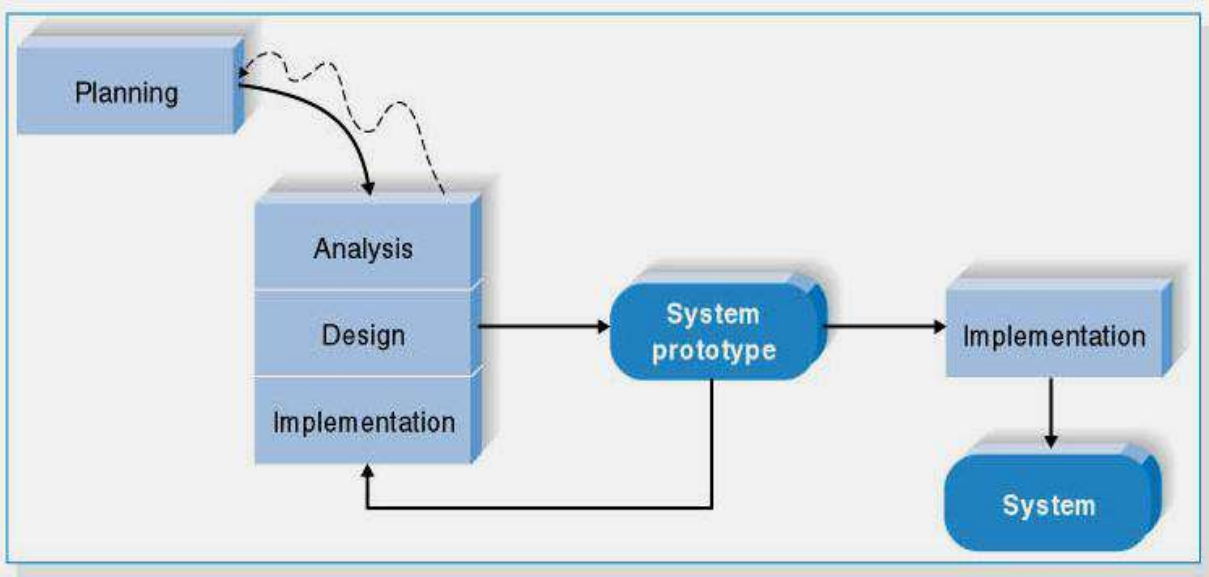
Phased Development A ***phased development***–based methodology breaks an overall system into a series of *versions*, which are developed sequentially. The analysis phase identifies the overall system concept, and the project team, users, and system sponsor then categorize the requirements into a series of versions. The most important and fundamental requirements are bundled into the first version of the system. The analysis phase then leads into design and implementation but only with the set of requirements identified for version 1. as shown above

Once version 1 is implemented, work begins on version 2. Additional analysis is performed based on the previously identified requirements and combined with new ideas and issues that arose from the users' experience with version 1. Version 2 then is designed and implemented, and work immediately begins on the next version. This process continues until the system is complete or is no longer in use.

Prototyping

A ***prototyping*-based** methodology performs the analysis, design, and implementation phases concurrently, and all three phases are performed repeatedly in a cycle until the system is completed. With these methodologies, the basics of analysis and design are performed, and work immediately begins on a *system prototype*, a "quick-and-dirty"

program that provides a minimal amount of features. The first prototype is usually the first part of the system that is used. This is shown to the users and the project sponsor, who pro- vide comments. These comments are used to reanalyze, redesign, and re-implement a second prototype, which provides a few more features. This process continues in a cycle until the analysts, users, and sponsor agree that the prototype provides enough functionality to be installed and used in the organization. After the prototype (now called the system) is installed, refinement occurs until it is accepted as the new system (see Figure below).

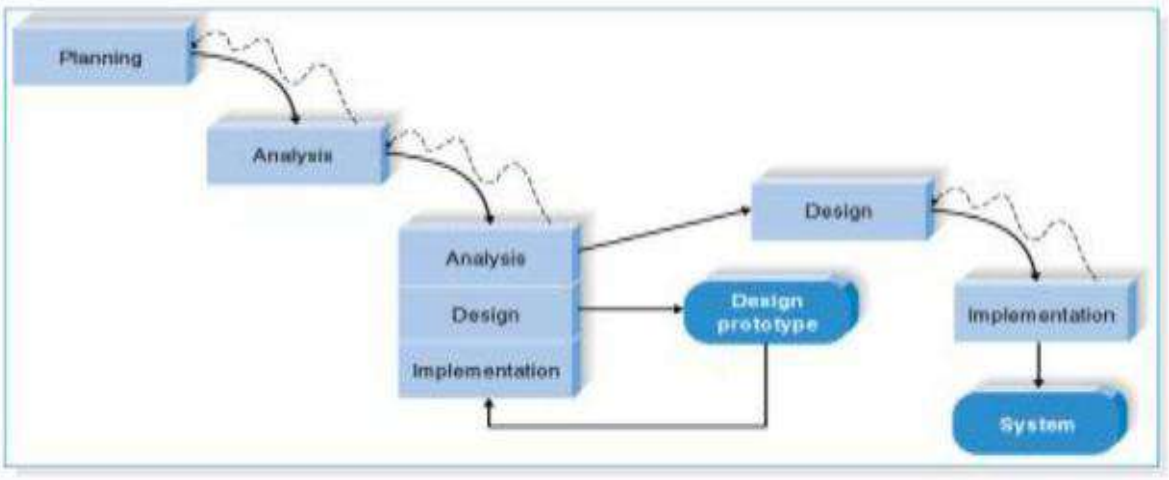


A Prototyping-based Methodology

- **Advantage**: Provides a system for the users to interact with, even if it is not initially ready for use.
- **Disadvantage**: Often the prototype undergoes such significant changes that many initial design decisions prove to be poor ones.

Throwaway Prototyping

Throwaway prototyping–based methodologies are similar to prototyping-based methodologies in that they include the development of prototypes; however, throwaway prototypes are done at a different point in the SDLC. These prototypes are used for a very different purpose than those previously discussed, and they have a very different appearance (see Figure below).



A Throwaway Prototyping–based Methodology

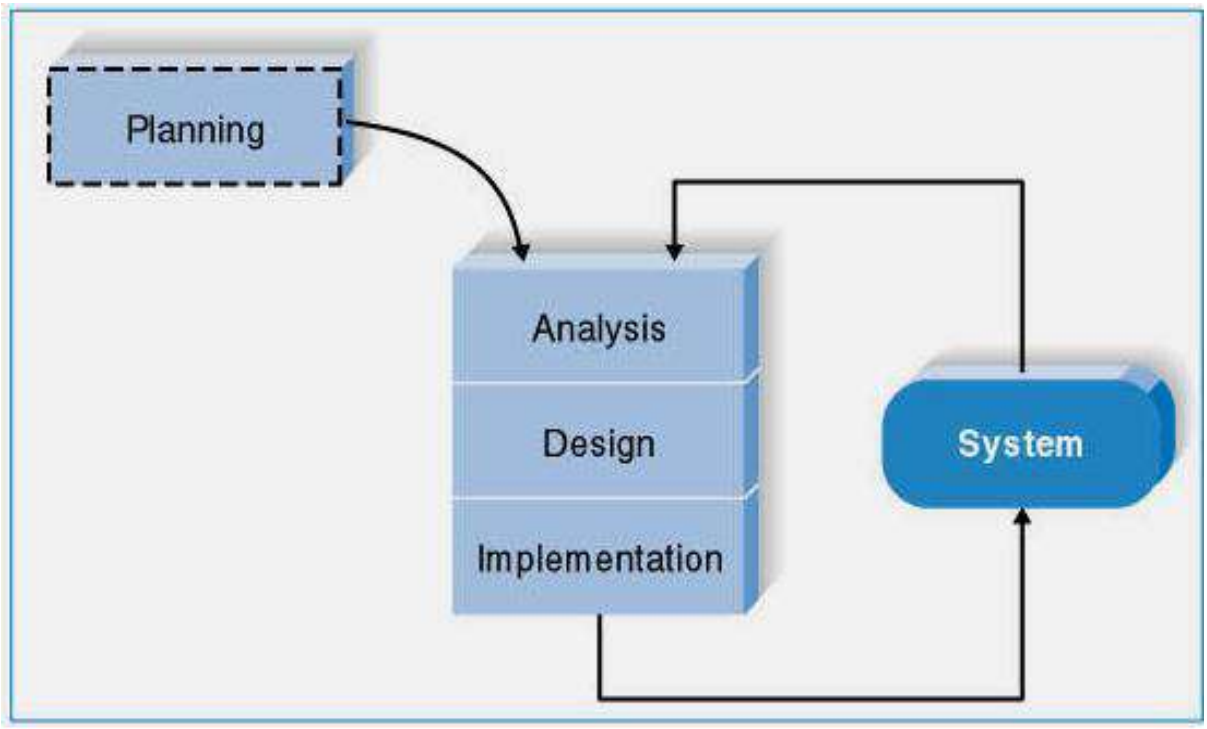
Agile Development

A third category of systems development methodologies is still emerging today: *agile development.* These programming-centric methodologies have few rules and practices, all of which are fairly easy to follow. They focus on streamlining the SDLC by eliminating much of the modeling and documentation overhead and the time spent on those tasks. Instead, projects emphasize simple, iterative application development. Examples of agile development methodologies include extreme programming, Scrum, and the Dynamic Systems Development Method (DSDM). The agile development approach, as described next, typically is used in conjunction with object-oriented methodologies.

Extreme Programming

Extreme programming (*XP*) is founded on four core values: communication, simplicity, feedback, and courage. These four values provide a foundation that XP developers use to create any system. First, the developers must provide rapid feedback to the end users on a continuous basis. Second, XP requires developers to follow the KISS principle. Third, developers must make incremental changes to grow the system, and they must not only accept change, they must embrace change. Fourth, developers must have a quality-first mentality. XP also supports team members in developing their own skills.

Three of the key principles that XP uses to create successful systems are continuous testing, simple coding performed by pairs of developers, and close interactions with end users to build systems very quickly. After a superficial planning process, projects perform analysis, design, and implementation phases iteratively (see figure Below).



The Extreme Programming Methodology

Selecting the Appropriate Development Methodology:

Selecting a methodology is not simple, as no one methodology is always best.
Many organizations have their own standards.
The next figure summarizes some important methodology selection criteria.

Ability to Develop Systems	Structured Methodologies		RAD Methodologies			Agile Methodologies
	Waterfall	Parallel	Phased	Prototyping	Throwaway Prototyping	XP
with Unclear User Requirements	Poor	Poor	Good	Excellent	Excellent	Excellent
with Unfamiliar Technology	Poor	Poor	Good	Poor	Excellent	Poor
that are Complex	Good	Good	Good	Poor	Excellent	Poor
that are Reliable	Good	Good	Good	Poor	Excellent	Good
with a Short Time Schedule	Poor	Good	Excellent	Excellent	Good	Excellent
with Schedule Visibility	Poor	Poor	Excellent	Excellent	Good	Good

Clarity of user Requirements:

RAD methodologies of prototyping and throwaway prototyping are usually more appropriate when user requirements are unclear as they provide prototypes for users to interact with early in the SDLC.

Familiarity with Technology:

If the system is designed without some familiarity with the base technology, risks increase because the tools may not be capable of doing what is needed.

System complexity:

Complex systems require careful and detailed analysis and design.
Project teams who follow phased development-based methodologies tend to devote less attention to the analysis of the complete problem domain than they might if they were using other methodologies.

System Reliability:

System reliability is usually an important factor in system development.

Throwaway prototyping-based methodologies are most appropriate when

system reliability is a high priority. Prototyping-based methodologies are generally not a good choice as they lack careful analysis and design phases.

Short Time Schedules:

RAD-based methodologies are well suited for projects with short time schedules as they increase speed.

Waterfall-based methodologies are the worst choice when time is essential as they do not allow for easy schedule changes.

Schedule Visibility:

RAD-based methodologies move many of the critical design decisions earlier in the project; consequently, this helps project managers recognize and address risk factors and keep expectations high.

Project team Skills and Roles:

Projects should consist of a variety of skilled individuals in order for a system to be successful.

Six major skill sets an analyst should have include:

- Technical
- Business
- Analytical
- Interpersonal
- Management
- Ethical

Categories of Analysts:

- Business Analyst
- Systems Analyst
- Infrastructure Analyst
- Change Management Analyst
- Project Manager

Project Team Roles

Role	Responsibilities
Business analyst	Analyzing the key business aspects of the system
	Identifying how the system will provide business value
	Designing the new business processes and policies
Systems analyst	Identifying how technology can improve business processes
	Designing the new business processes
	Designing the information system
	Ensuring that the system conforms to information systems standards
Infrastructure analyst	Ensuring the system conforms to infrastructure standards
	Identifying infrastructure changes needed to support the system
Change management analyst	Developing and executing a change management plan
	Developing and executing a user training plan
Project manager	Managing the team of analysts, programmers, technical writers, and other specialists
	Developing and monitoring the project plan
	Assigning resources
	Serving as the primary point of contact for the project

INITIATION:

This chapter describes Project Initiation, the point at which an organization creates and assesses the original goals and expectations for a new system. The first step in the process is to identify a project that will deliver value to the business and to create a system request that provides basic information about the proposed system. Next, the analysts perform a feasibility analysis to determine the technical, economic, and organizational feasibility of the system; if appropriate, the system is selected and the development project begins.

INTRODUCTION

The first step in any new development project is for someone a manager, staff member, sales representative, or systems analyst to see an opportunity to improve the business. New systems start first and foremost from a business need or opportunity. Many ideas for new systems or improvements to existing ones arise from the application of a new technology, but an understanding of technology is usually secondary to a solid understanding of the business and its objectives.

Project Identification

A project is identified when someone in the organization identifies a *business need* to build a system. This could occur within a business unit or IT, come from a steering committee charged with identifying business opportunities, or evolve from a recommendation made by external consultants. Examples of business needs include supporting a new marketing campaign, reaching out to a new type of customer, or improving interactions with suppliers. Sometimes, needs arise from some kind of "pain" within the organization, such as a drop in market share, poor customer service levels, or increased competition. Other times, new business initiatives and strategies are created, and a system is required to enable them.

Business needs also can surface when the organization identifies unique and competitive ways of using IT. Many organizations keep an eye on ***emerging technology***, which is technology that is still being developed and is not yet viable for widespread business use. For example, if companies stay abreast of technology such as the Internet, smart cards, and scent-technology in their earliest stages, they can develop business strategies that leverage the capabilities of these technologies and introduce them into the marketplace as a *first mover*. Ideally, they can take advantage of this first-mover advantage by making money and continuing to innovate while competitors trail behind.

System Request

A system request is a document that describes the business reasons for building a sys- tem and the value that the system is expected to provide. The project sponsor usually completes this form as part of a formal system project selection process within the organization. Most system requests include five elements: project sponsor, business need, business requirements, business value, and *special issues* (see Figure below). The sponsor describes the person who will serve as the primary contact for the project, and the business need presents the reasons prompting the project. The business requirements of the project refer to the business capabilities that the system will need to have, and the business value describes the benefits that the organization should expect from the system. Special issues are included on the document as a catch-all for other information that should be considered in assessing the project. For example, the project may need to be completed by a specific deadline. Project teams need to be aware of any special circumstances that could affect the outcome of the system. Figure below shows a template for a system request.

Feasibility Analysis

Once the need for the system and its business requirements have been defined, it is time to create a more detailed business case to better understand the opportunities and limitations associated with the proposed project. Feasibility analysis guides the organization in determining whether or not to proceed with a project. Feasibility analysis also identifies the important *risks* associated with the project that must be addressed if the project is approved. As with the system request, each organization has its own process and format for the feasibility analysis, but most include three techniques: technical feasibility, economic feasibility, and organizational feasibility. The results of these techniques are combined into a *feasibility study* deliverable, which is given to the approval committee at the end of project initiation.

Economic Feasibility

The second element of a feasibility analysis is to perform an *economic feasibility* analysis (also called a *cost–benefit analysis*), which identifies the financial risk associated with the project. It attempts to answer this question: *Should* we build the system? Economic feasibility is determined by identifying costs and benefits associated with the system, assigning values to them, and then calculating the cash flow and return on investment for the project. The more expensive the project, the more rigorous and detailed the analysis should be.

Identifying Costs and Benefits

The first task when developing an economic feasibility analysis is to identify the kinds of costs and benefits the system will have and list them along the left-hand column of a spreadsheet. lists examples of costs and benefits that may be included.

Costs and benefits can be broken down into four categories:

(1) development costs, (2) operational costs, (3) tangible benefits, and (4) intangibles.

Development costs are those tangible expenses incurred during the construction of the system, such as salaries for the project team, hardware and software expenses, consultant fees, training, and office space and equipment. Development costs are

usually thought of as one-time costs. ***Operational costs*** are those tangible costs required to operate the system, such the salaries for operations staff, software licensing fees, equipment upgrades, and communications charges. Operational costs are usually thought of as ongoing costs.

Tangible vs. Intangible costs:

- **Tangible Costs** – Includes revenue that the system enables the organization to collect, such as increased sales.
- **Intangible Costs** – Are based on intuition and belief rather than "hard numbers."

Assigning Values to Costs and Benefits

Once the types of costs and benefits have been identified, analysts assign specific dollar values to them. This may seem impossible; how can someone quantify costs and benefits that haven't happened yet? And how can those predictions be realistic? Although this task is very difficult, they have to do the best they can to come up with reasonable numbers for all the costs and benefits. Only then can the approval committee make an educated decision about whether or not to move ahead with the project.

Organizational Feasibility

The final technique used for feasibility analysis is to assess the ***organizational feasibility*** of the system, how well the system ultimately will be accepted by its users and incorporated into the ongoing operations of the organization. There are many organizational factors that can have an impact on the project, and seasoned developers know that organizational feasibility can be the most difficult feasibility dimension to assess. In essence, an organizational feasibility analysis attempts to answer this question: If we build it, will they come?

One way to assess the organizational feasibility of the project is to understand how well the goals of the project align with business objectives. *Strategic alignment* is the fit between the project and business strategy the greater the alignment, the less risky the project will be from an organizational feasibility perspective. For example, if the marketing department has decided to become more customer focused, then a CRM project that produces integrated customer information would have strong strategic

alignment with marketing's goal. Many IT projects fail when the IT department initiates them, because there is little or no alignment with business unit or organizational strategies.

Project Selection

The approval committee met and reviewed the Internet Sales System project along with two other projects one calling for the implementation of a corporate Intranet and another proposing in store kiosks that would provide customers with information about the CDs that the store carried. Unfortunately, the budget would allow for only one project to be approved, so the committee carefully examined the costs, expected benefits, risks, and strategic alignment of all three projects. Currently, a primary focus of upper management is increasing sales in the retail stores and the Internet system and kiosk project best aligned with that goal. Given that both projects had equal risk but that the Internet Order project expected a much greater return, the committee decided to fund the Internet Sales System.

Project Management:

This chapter describes the important steps of project management, which begins in planning and continues throughout the SDLC. First, the project manager estimates the size of the project and identifies the tasks that need to be performed. Next, he or she staffs the project and puts several activities in place to help coordinate project activities. These steps produce important project management deliverables, including the work plan, staffing plan, and standards list.

INTRODUCTION

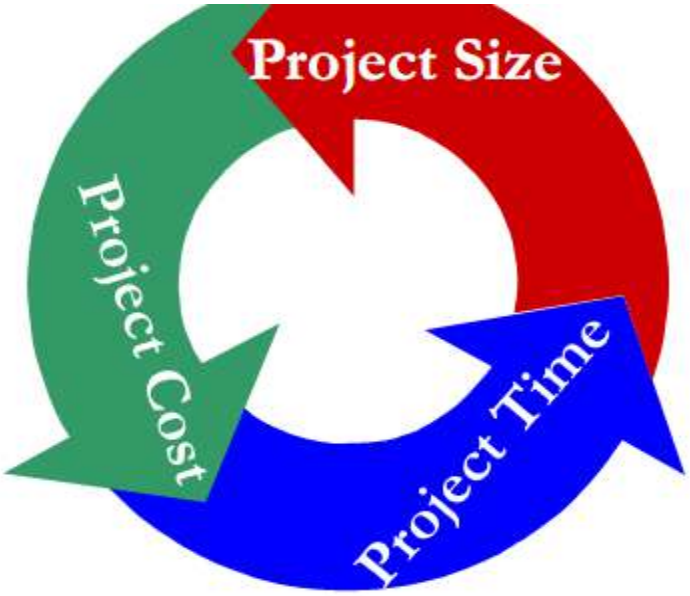
Project management is the process of planning and controlling the development of a system within a specified time frame at a minimum cost with the right functionality. A **project manager** has the primary responsibility for managing the hundreds of tasks and roles that need to be carefully coordinated. Today, project management is an actual profession, and analysts spend years working on projects prior to tackling the management of them. In a 1999 *Computerworld* survey, more than half of 103 companies polled said they now offer formal project management training for information technology (IT) project teams. There also is a variety of *project management software* available, such as Microsoft Project, Plan View, and PM Office, which support project management activities.

Four Key Steps in Managing projects:

- Identifying project size
- Creating and managing the work plan
- Staffing the project
- Coordinating project activities

Identifying Project Size:

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



Project Estimation:

Therefore, in the beginning of the project, the manager needs to estimate each of these levers and then continuously assess how to roll out the project in a way that meets the organization's needs. ***Estimation*** is the process of assigning projected values for time and effort, and it can be performed manually or with the help of estimation software packages such as Costar and Construx there are more than fifty available on the market. The estimates developed at the start of a project are usually based on a range of possible values (e.g., the design phase will take three to four months) and gradually become more specific as the project moves forward (e.g., the design phase will be completed on March 22).

Project time using the Planning Phase Approach:

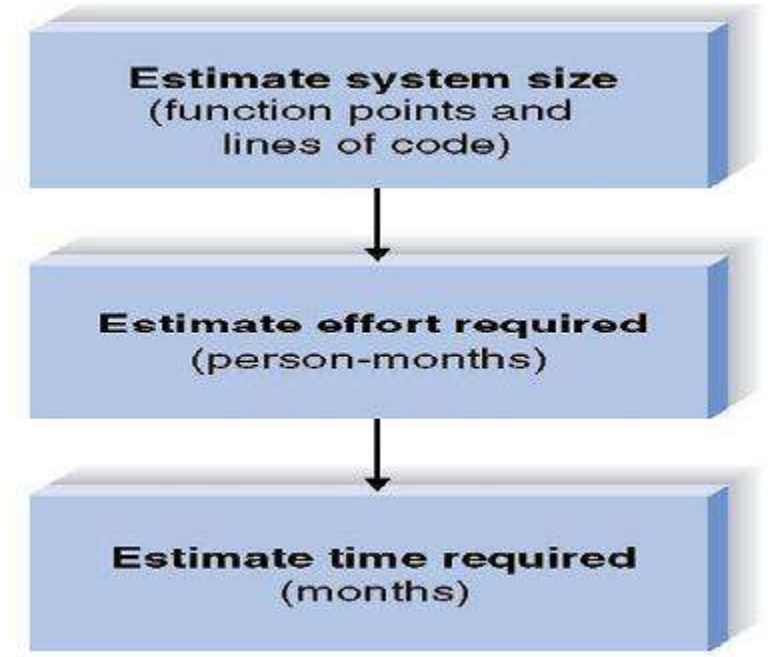
	Planning	Analysis	Design	Implementation
Typical industry standards for business applications	15%	20%	35%	30%
Estimates based on actual figures for first stages of SDLC	Actual: 4 person-months	Estimated: 5.33 person-months	Estimated: 9.33 person-months	Estimated: 8 person-months

SDLC = systems development life cycle.

PROJECT ESTIMATION USING THE FUNCTION POINT APPROACH

Function Point Approach

The *function point approach* to estimation uses a more complex—and, it is hoped, more reliable three-step process (see Figure below2). First, the project manager estimates the size of the project in terms of the number of lines of code the new system will require. This size estimate is then converted into the amount of effort required to develop the system in terms of a number of person- months. The estimated effort is then converted into an estimated schedule time in terms of the number of months from start to finish.



Creating and managing the work plan

Once a project manager has a general idea of the size and approximate schedule for the project, he or she creates a ***work plan***, which is a dynamic schedule that records and keeps track of all the tasks that need to be accomplished over the course of the project. The work plan lists each task, along with important information about it, such as when it needs to be completed, the person assigned to do the work, and any

deliverables that will result (Figure shown below). The level of detail and the amount of information captured by the work plan depend on the needs of the project (and the detail usually increases as the project progresses). The work plan is usually the main component of the project management software mentioned earlier.

Work Plan Information	Example
Name of task	Perform economic feasibility
Start date	Jan 05, 2005
Completion date	Jan 19, 2005
Person assigned	Project sponsor: Mary Smith
Deliverable(s)	Cost-benefit analysis
Completion status	Open
Priority	High
Resources needed	Spreadsheet
Estimated time	16 hours
Actual time	14.5 hours

Work Plan Example

Identifying Tasks

The overall objectives for the system should be listed on the system request, and it is the project manager's job to identify all the tasks that need to be accomplished to meet those objectives. This sounds like a daunting task how can someone know everything that needs to be done to build a system that has never been built before?

The Project Work plan

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

Gantt Chart

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

Pert Chart

A second graphical way to look at project work plan information is the *PERT chart*, which lays out the project tasks in a flowchart. PERT, which stands for program evaluation and review technique, is a network analysis technique that can be used when the individual task time estimates are fairly uncertain. Instead of simply putting a point estimate for the duration estimate, PERT uses three time estimates: optimistic, most likely, and a pessimistic. It then combines the three estimates into a single weighted average estimate using the following formula:

Refining Estimates

The estimates that are produced during planning need to be refined as the project progresses. This does not mean that estimates were poorly done at the start of the project; rather, it is virtually impossible to develop an exact assessment of the project's schedule before the analysis and design phases are conducted. A project

manager should expect to be satisfied with broad ranges of estimates that become more and more specific as the project's product becomes better defined.

Managing scope:

- Scope creep
- JAD and prototyping
- Formal change approval
- Defer additional requirements as future system enhancements

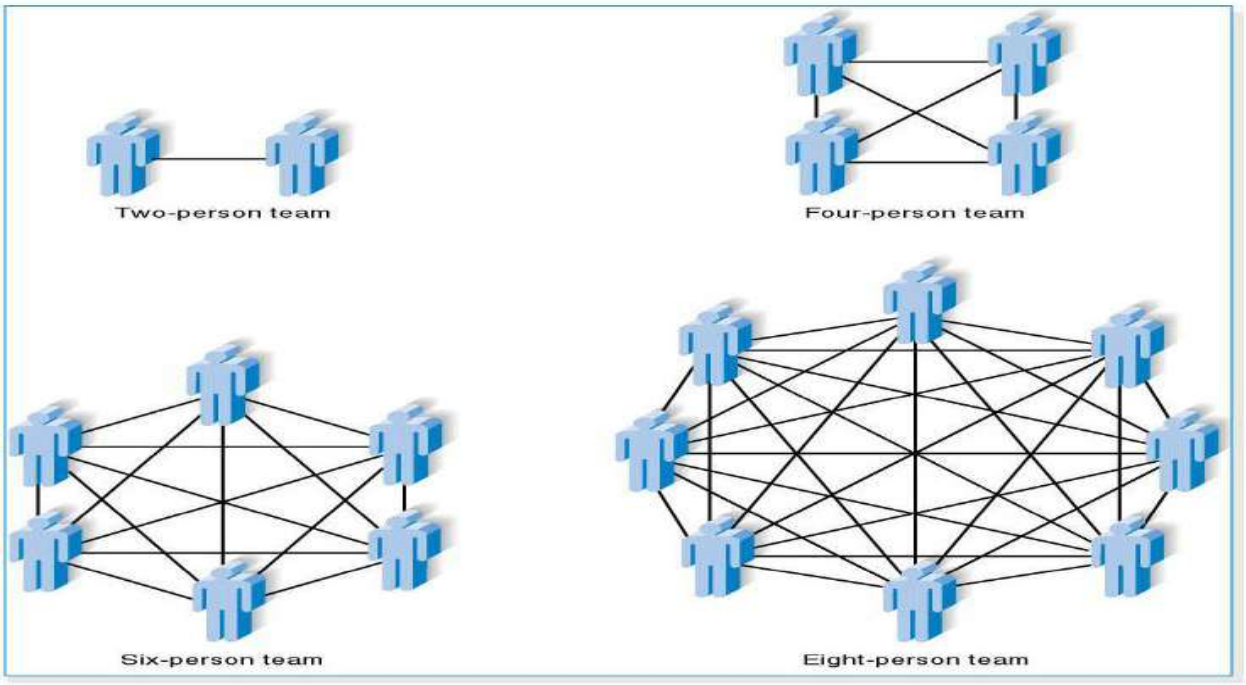
Time boxing:

- Fixed deadline
- Reduced functionality, if necessary
- Fewer "finishing touches"

Staffing the Project

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

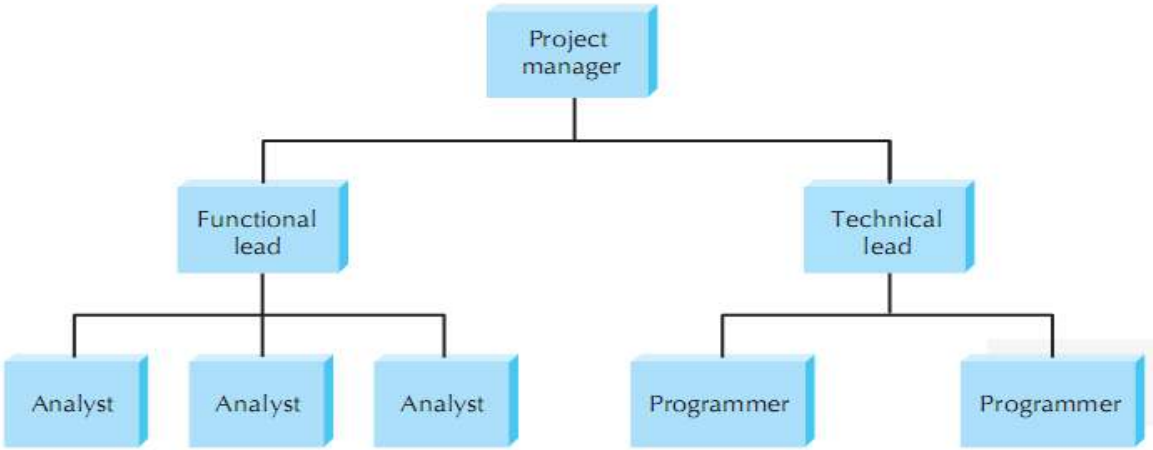
- Staffing levels will change over a project's lifetime
- Adding staff may add more overhead than additional labor
- Using teams of 8-10 reporting in a hierarchical structure can reduce complexity



Increasing complexity with larger teams

Key Definitions:

- The **staffing plan** describes the kinds of people working on the project
- The **project charter** describes the projects objectives and rules
- A **functional lead** manages a group of analysts
- A **technical lead** oversees progress of programmers and technical staff members



Motivation

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

Handling Conflict

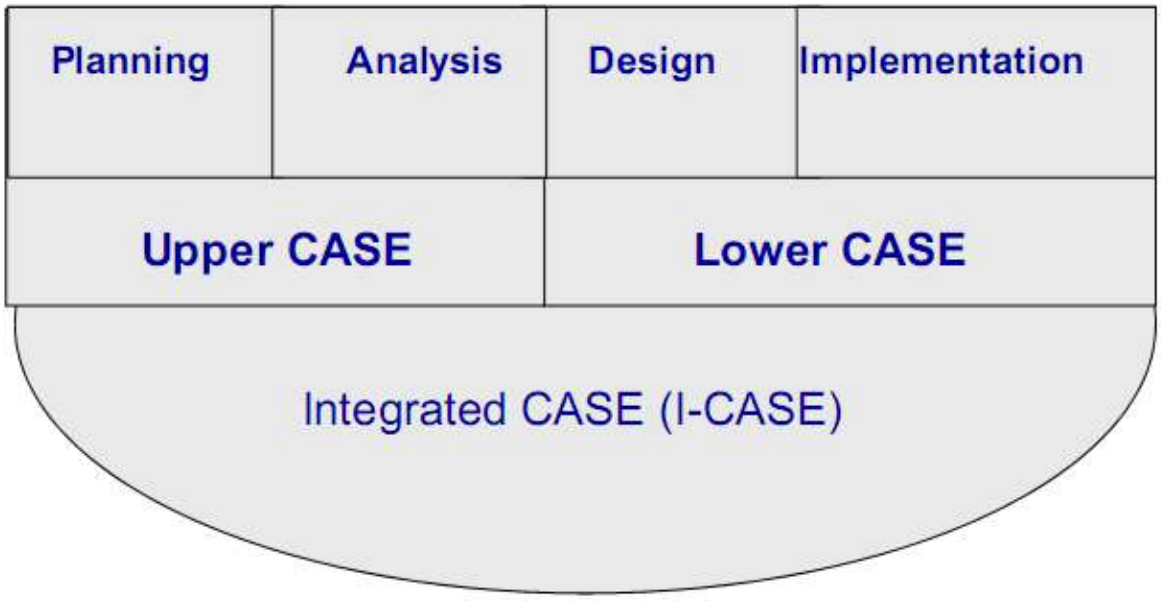
The third component of staffing is organizing the project to minimize conflict among group members. *Group cohesiveness* (the attraction that members feel to the group and to other members) contributes more to productivity than do project members' individual capabilities or experiences. Clearly defining the roles on the project and holding team members accountable for their tasks is a good way to begin mitigating potential conflict on a project.

COORDINATING PROJECT ACTIVITIES:

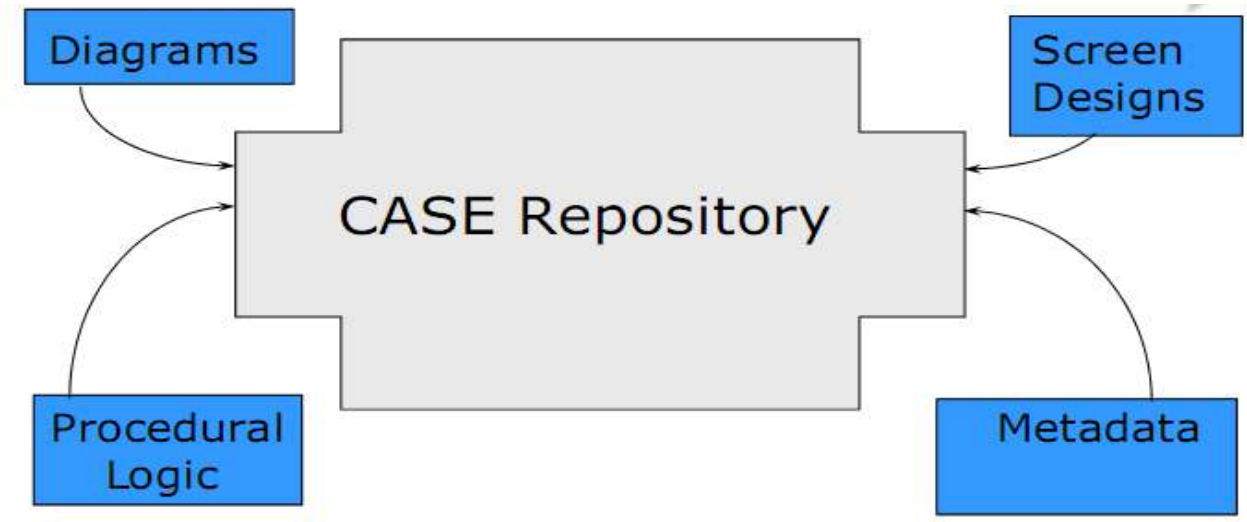
Case Tools

Computer-aided software engineering (CASE) is a category of software that automates all or part of the development process. Some CASE software packages are used primarily during the analysis phase to create integrated diagrams of the system and to store information regarding the system components (often called *upper CASE*), whereas others are design phase tools that create the diagrams and then generate code for database tables and system functionality (often called *lower CASE*). *Integrated CASE*, or I-CASE, contains functionality found in both upper

CASE and lower CASE tools in that it supports tasks that happen throughout the SDLC. CASE comes in a wide assortment of flavors in terms of complexity and functionality, and there are many good programs available in the marketplace (e.g., Visible Analyst Workbench, Oracle Designer/2000, Rational Rose, Logic Works suite).



Case Tools



Case Components

Standards

Members of a project team need to work together, and most project management soft- ware and CASE tools provide access privileges to everyone working on the system. When people work together, however, things can get pretty confusing. To make matters worse, people sometimes are reassigned in the middle of a project. It is important that their project knowledge does not leave with them and that their replacements can get up to speed quickly.

Documentation

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

Managing Risk

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

REQUIREMENTS DETERMINATION:

Introduction

One of the first activities of an analyst is to determine the business requirements for a new system. This chapter begins by presenting the requirements definition, a document that lists the new system's capabilities. It then describes how to analyze requirements using business process automation, business process improvement, and business process reengineering techniques and how to gather requirements using interviews, JAD sessions, questionnaires, document analysis, and observation.

Defining a Requirement

A *requirement* is simply a statement of what the system must do or what characteristic it must have. During analysis, requirements are written from the perspective of the businessperson, and they focus on the "what" of the system. Because they focus on the needs of the business user, they are usually called *business requirements* (and sometimes user requirements). Later in design, business requirements evolve to become more technical, and they describe how the system will be implemented. Requirements in design are written from the developer's perspective, and they are usually called ***system requirements.***

Creating a Requirements Definition

Creating a requirements definition is an iterative and ongoing process whereby the analyst collects information with requirements-gathering techniques (e.g., interviews, document analysis), critically analyzes the information to identify appropriate business requirements for the system, and adds the requirements to the requirements definition report. The requirements definition is kept up to date so that the project team and business users can refer to it and get a clear understanding of the new system.

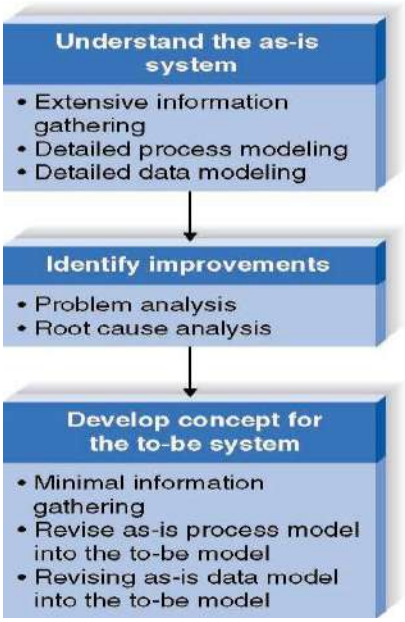
To create a requirements definition, the project team first determines the kinds of functional and nonfunctional requirements that they will collect about the system (of course, these may change over time). These become the main sections of the document. Next, the analysts use a variety of requirements gathering techniques (e.g., interviews, observation) to collect information, and they list the business requirements that were identified from that information. Finally, the analysts work with the entire project team and the business users to verify, change, and complete the list and to help prioritize the importance of the requirements that were identified.

Requirements Analysis Strategies

Before the project team can determine what requirements are appropriate for a given sys- tem, they need to have a clear vision of the kind of system that will be created and the level of change that it will bring to the organization. The basic process of *analysis* is divided into three steps: understanding the as-is system, identifying improvements, and developing requirements for the to-be system.

Business Process Automation

BPA leaves the basic way in which the organization operates unchanged and uses computer technology to do some of the work. BPA can make the organization more efficient but has the least impact on the business. Planners in BPA projects spend a significant time understanding the current as is system before moving on to improvements and to be system requirements. Problem analysis and root cause analysis are two popular BPA techniques.



Technique: Business Process Automation

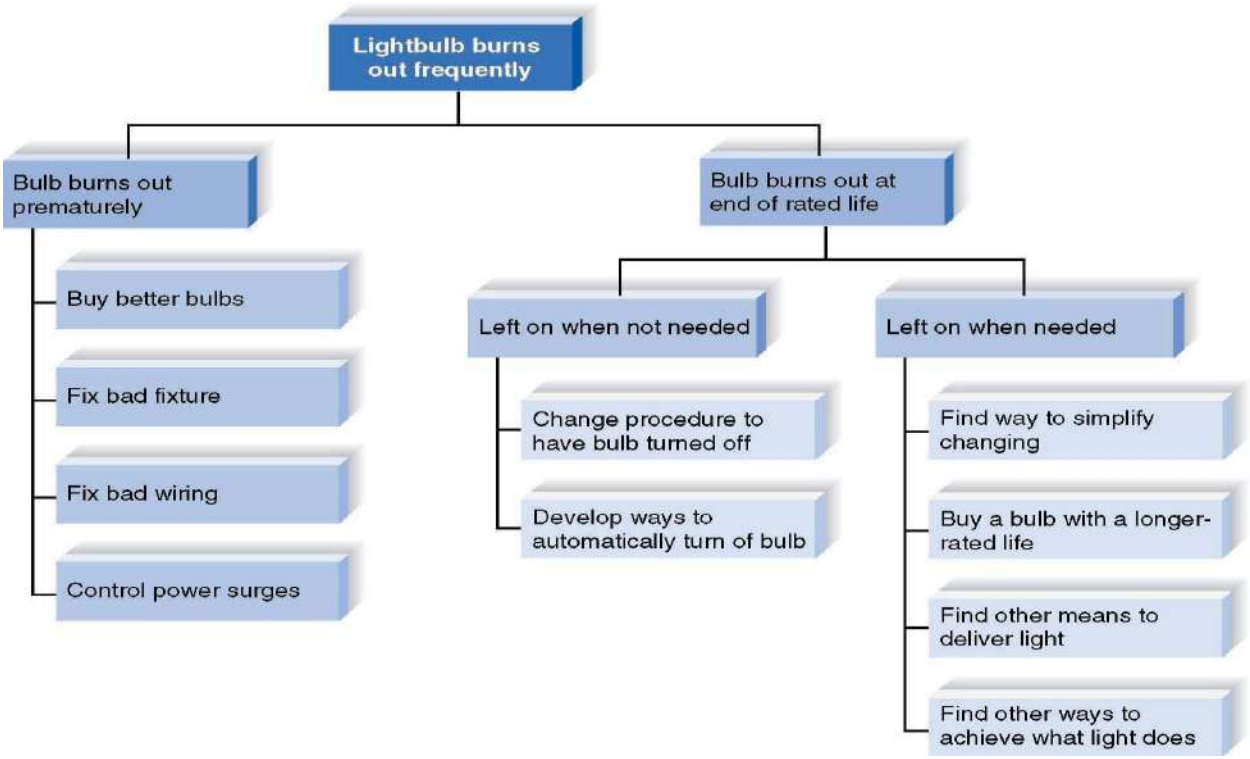
Problem Analysis

The most straightforward (and probably the most commonly used) requirements analysis technique is *problem analysis*. Problem analysis means asking the users and managers to identify problems with the as-is system and to describe how to solve them in the to-be system. Most users have a very good idea of the changes they would like to see, and most will be quite vocal about suggesting them. Most changes tend to solve problems rather than capitalize on opportunities, but the latter is possible as well. Improvements from problem analysis tend to be small and incremental (e.g., provide more space in which to type the customer's address; provide a new report that currently does not exist).

Root Cause Analysis

The ideas produced by problem analysis tend to be solutions to problems. All solutions make assumptions about the nature of the problem, assumptions that may or may not be valid. In our experience, users (and most people in general) tend to quickly jump to solutions without fully considering the nature of the problem.

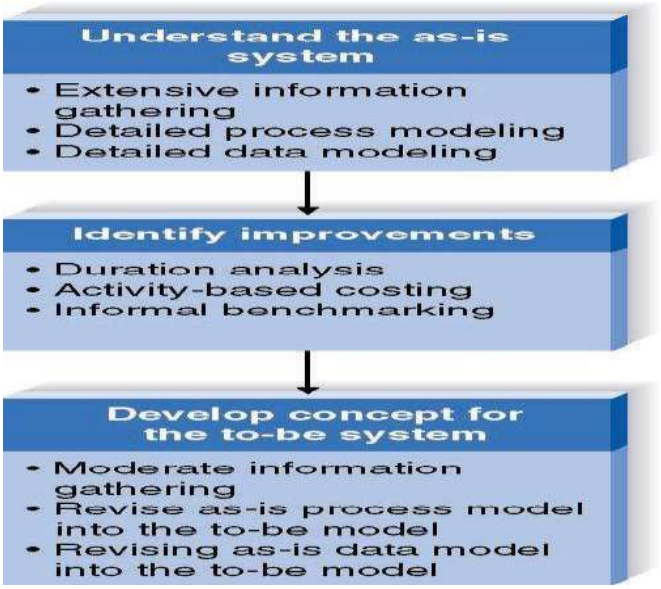
Sometimes the solutions are appropriate, but many times they address a *symptom* of the problem, not the true problem or *root cause* itself.



Root cause analysis example

Business Process Improvement

BPI makes moderate changes to the way in which the organization operates to take advantage of new opportunities offered by technology or to copy what competitors are doing. BPI can improve efficiency (i.e., doing things right) and improve effectiveness (i.e., doing the right things). Planners of BPI projects also spend time understanding the as-is system, but much less time than with BPA projects; their primary focus is on improving business processes, so time is spent on the as-is only to help with the improvement analyses and the to-be system requirements. *Duration analysis*, *activity-based costing*, and *informal bench-marking* are three popular BPI activities.



Business Process Improvement

1. **Duration Analysis:** Duration analysis requires a detailed examination of the amount of time it takes to perform each process in the current as-is system. The analysts begin by determining the total amount of time it takes, on average, to perform a set of business processes for a typical input. They then time each of the individual steps (or sub- processes) in the business process. The time to complete the basic steps are then totaled and compared to the total for the overall process. A significant difference between the two and in our experience the total time often can be 10 or even 100 times longer than the sum of the parts indicates that this part of the process is badly in need of a major overhaul.
2. **Activity-Based Costing:** Activity-based costing is a similar analysis, which examines the cost of each major process or step in a business process rather than the time taken.[5] The analysts identify the costs associated with each of the basic functional steps or processes, identify the most costly processes, and focus their improvement efforts on them.

3. **Informal Benchmarking:** Benchmarking refers to studying how other organizations perform a business process in order to learn how your organization can do something better. Benchmarking helps the organization by introducing ideas that employees may never have considered but have the potential to add value.

Business Process Reengineering

BPR means changing the fundamental way in which the organization operates, obliterating the current way of doing business and making major changes to take advantage of new ideas and new technology. Planners of BPR projects spend little time understanding the as-is, because their goal is to focus on new ideas and new ways of doing business. Outcome analysis, technology analysis, and activity elimination are three popular BPR activities.

1. **Outcome Analysis** *Outcome analysis* focuses on understanding the fundamental out- comes that provide value to customers. Although these outcomes sound as though they should be obvious, they often aren't. For example, suppose you consider an insurance company. One of its customers has just had a car accident. What is the fundamental outcome from the *customer's* perspective?

2. **Technology Analysis:** Many major changes in business over the past decade have been enabled by new technologies. *Technology analysis* starts by having the analysts and managers develop a list of important and interesting technologies. Then the group systematically identifies how each and every technology could be applied to the business process and identifies how the business would benefit.

3. **Activity Elimination:** *Activity elimination* is exactly what it sounds like. The analysts and managers work together to identify how the organization could eliminate each and every activity in the business process, how the function could operate without it, and what effects are likely to occur. Initially, managers are reluctant to conclude that processes can be eliminated, but this is a "force-fit" exercise in that they must eliminate each activity. In some cases the results are silly; nonetheless, participants must address each and every activity in the business process.

Selecting Appropriate Strategies

Each technique discussed in this chapter has its own strengths and weaknesses (see Figure below). No one technique is inherently better than the others, and in practice most projects use a combination of techniques.

	Business Process Automation	Business Process Improvement	Business Process Reengineering
Potential business value	Low–moderate	Moderate	High
Project cost	Low	Low–moderate	High
Breadth of analysis	Narrow	Narrow–moderate	Very broad
Risk	Low–moderate	Low–moderate	Very high

1. **Potential Business Value:** *Potential business value* varies with the analysis strategy. Although BPA has the potential to improve the business, most of the benefits from BPA are tactical and small in nature. Because BPA does not seek to change the business processes, it can only improve their efficiency. BPI usually offers moderate potential benefits, depending upon the scope of the project, because it seeks to change the business in some way. It can increase both efficiency and effectiveness. BPR creates large

potential benefits because it seeks to radically improve the nature of the business.

2. **Project Cost:** *Project cost* is always important. In general, BPA has the lowest cost because it has the narrowest focus and seeks to make the fewest number of changes. BPI can be moderately expensive, depending upon the scope of the project. BPR is usually expensive, both because of the amount of time required of senior managers and the amount of redesign to business processes.

3. **Breadth of Analysis:** *Breadth of analysis* refers to the scope of analysis, or whether analysis includes business processes within a single business function, processes that cross the organization, or processes that interact with those in customer or supplier organizations. BPR takes a broad perspective, often spanning several major business processes, even across multiple organizations. BPI has a much narrower scope that usually includes one or several business functions. BPA typically examines a single process.

4. **Risk:** One final issue is *risk* of failure, which is the likelihood of failure due to poor design, unmet needs, or too much change for the organization to handle. BPA and BPI have low to moderate risk because the to-be system is fairly well defined and understood, and its potential impact on the business can be assessed before it is implemented. BPR projects, on the other hand, are less predictable. BPR is extremely risky and is not some- thing to be undertaken unless the organization and its senior leadership are committed to making significant changes. Mike Hammer, the father of BPR, estimates that 70 percent of BPR projects fail.

Requirements-Gathering Techniques

1. **Interview**: An interview is the most commonly used requirements-gathering technique. After all, it is natural if you need to know something, you usually ask someone. In general, interviews are conducted one-on-one (one interviewer and one interviewee), but sometimes, due to time constraints, several people are interviewed at the same time. There are five basic steps to the interview process: selecting interviewees, designing interview

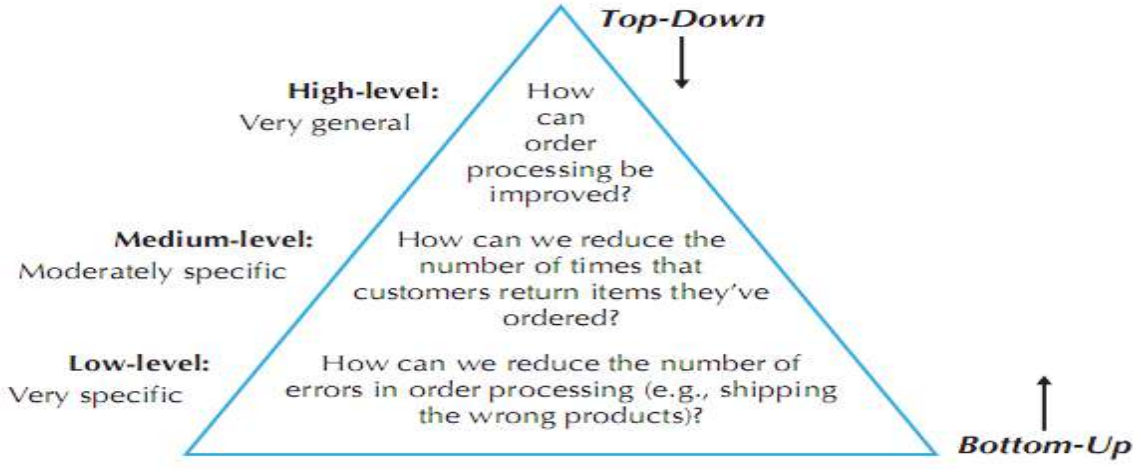
questions, preparing for the interview, conducting the interview, and post interview follow-up.

2. **Selecting Interviewees:** The first step in interviewing is to create an *interview schedule* listing all the people who will be interviewed, when, and for what purpose. The schedule can be an informal list that is used to help set up meeting times or a formal list that is incorporated into the work plan. The people who appear on the interview schedule are selected based on the analyst's information needs. The project sponsor, key business users, and other members of the project team can help the analyst determine who in the organization can best provide important information about requirements. These people are listed on the interview schedule in the order in which they should be interviewed.
3. **Designing Interview Questions:** There are three types of interview questions: closed- ended questions, open-ended questions, and probing questions. *Closed-ended questions* are those that require a specific answer. They are similar to multiple-choice or arithmetic questions on an exam (see Figure below). Closed-ended questions are used when an analyst is looking for specific, precise information (e.g., how many credit card requests are received per day). In general, precise questions are best. For example, rather than asking, Do you handle a lot of requests? it is better to ask, How many requests do you process per day?

   Closed-ended questions enable analysts to control the interview and obtain the information they need. However, these types of questions don't uncover *why* the answer is the way it is, nor do they uncover information that the interviewer does not think to ask ahead of time.

Types of Questions	Examples
Closed-ended questions	• How many telephone orders are received per day? • How do customers place orders? • What information is missing from the monthly sales report?
Open-ended questions	• What do you think about the current system? • What are some of the problems you face on a daily basis? • What are some of the improvements you would like to see in a new system?
Probing questions	• Why? • Can you give me an example? • Can you explain that in a bit more detail?

The top-down approach is an appropriate strategy for most interviews (it is certainly the most common approach). The top-down approach enables the interviewee to become accustomed to the topic before he or she needs to provide specifics. It also enables the interviewer to understand the issues before moving to the details because the interviewer may not have sufficient information at the start of the interview to ask very specific questions. Perhaps most importantly, the top-down approach enables the interviewee to raise a set of big-picture issues before becoming enmeshed in details, so the interviewer is less likely to miss important issues.



Top-Down and Bottom-Up Questioning Strategies

Preparing for the Interview It is important to prepare for the interview in the same way that you would prepare to give a presentation. The interviewer should have a general interview plan listing the questions to be asked in the appropriate order; should anticipate possible answers and provide follow-up with them; and should identify segues between related topics. The interviewer should confirm the areas in which the interviewee has knowledge in order not to ask questions that he or she cannot answer. Review the topic areas, the questions, and the interview plan, and clearly decide which have the greatest priority in case time runs short.

Conducting the Interview In starting the interview, the first goal is to build rapport with the interviewee, so that he or she trusts the interviewer and is willing to tell the whole truth, not just give the answers that he or she thinks are wanted. The interviewer should appear to be professional and an unbiased, independent seeker of information. The interview should start with an explanation of why the interviewer is there and why he or she has chosen to interview the person; then the interviewer should move into the planned interview questions.

Post interview Follow-up After the interview is over, the analyst needs to prepare an *interview report* that describes the information from the interview. The report contains *interview notes,* information that was collected over the course of the interview and is summarized in a useful format. In general, the interview report should be written within forty-eight hours of the interview, because the longer the interviewer waits, the more likely he or she is to forget information.

Joint Application Development (JAD)

JAD is an information-gathering technique that allows the project team, users, and management to work together to identify requirements for the system. IBM developed the JAD technique in the late 1970s, and it is often the most useful method for collecting information from users. Capers Jones claims that JAD can reduce scope creep by 50 percent, and it avoids the requirements for a system being too specific or too vague, both of which cause trouble during later stages of the SDLC. JAD is a structured process in which ten to twenty users meet together under the direction of a *facilitator* skilled in JAD techniques. The facilitator is a person who sets the meeting agenda and guides the discussion but does not join in the discussion as a participant. He or she does not provide ideas or opinions on the topics under discussion in order to remain neutral during the session. The facilitator must be an expert in both group process techniques and systems analysis and design techniques. One or two *scribes* assist the facilitator by recording notes, making copies, and so on. Often the scribes will use computers and CASE tools to record information as the JAD session proceeds.

1. **Selecting Participants** Selecting JAD participants is done in the same basic way as selecting interview participants. Participants are selected based on the information they can con- tribute, to provide a broad mix of organizational levels, and to build political support for the new system. The need for all JAD participants to be away from their office at the same time can be a major problem. The office may need to be closed or operate with a skeleton staff until the JAD sessions are complete.

2. **Designing a JAD Session** JAD sessions can run from as little as half a day to several weeks, depending upon the size and scope of the project. In

our experience, most JAD sessions tend to last five to ten days, spread over a three-week period. Most e-JAD sessions tend to last one to four days in a one-week period. JAD and e-JAD sessions usually go beyond the collection of information and move into analysis. For example, the users and the analysts collectively can create analysis deliverables, such as the functional models, structural models, or the requirements definition.

3. **Preparing for a JAD Session** As with interviewing, it is important to prepare the analysts and participants for a JAD session. Because the sessions can go beyond the depth of a typical interview and are usually conducted off-site, participants can be more concerned about how to prepare. It is important that the participants understand what is expected of them. If the goal of the JAD session, for example, is to develop an understanding of the current system, then participants can bring procedure manuals and documents with them. If the goal is to identify improvements for a system, then they can think about how they would improve the system prior to the JAD session.

4. **Conducting a JAD Session** Most JAD sessions try to follow a formal agenda, and most have formal *ground rules* that define appropriate behavior. Common ground rules include following the schedule, respecting others' opinions, accepting disagreement, and ensuring that only one person talks at once.

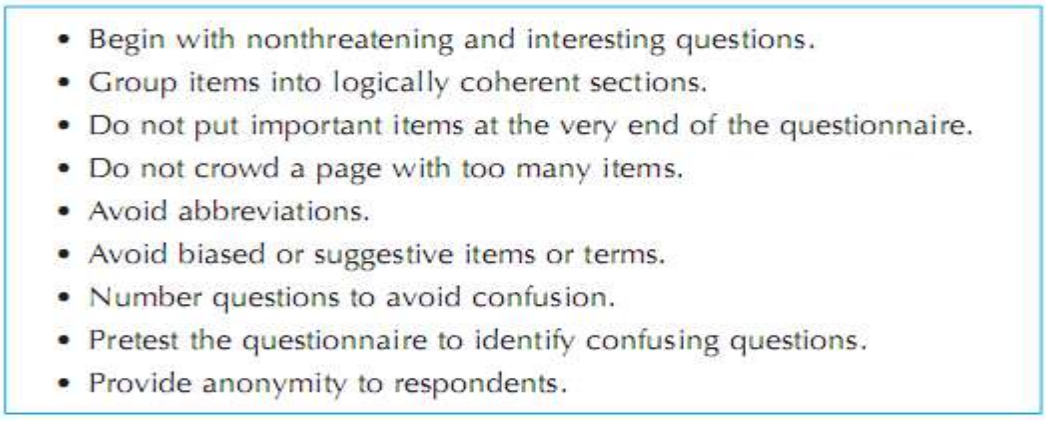
   The role of a JAD facilitator can be challenging. Many participants come to a JAD session with strong feelings about the system to be discussed. Channeling these feelings so that the session moves forward in a positive direction and getting participants to recognize and accept but not necessarily agree on—opinions and situations different from their own

requires significant expertise in systems analysis and design, JAD, and interpersonal skills. Few systems analysts attempt to facilitate JAD sessions without being trained in JAD techniques, and most apprentice with a skilled JAD facilitator before they attempt to lead their first session.

5. **Post-JAD Follow-up:** As with interviews, a JAD *post session report* is prepared and circulated among session attendees. The post session report is essentially the same as the interview report. Because the JAD sessions are longer and provide more information, it usually takes a week or two after the JAD session before the report is complete.

Questionnaires

A questionnaire is a set of written questions used to obtain information from individuals. Questionnaires are often used when there is a large number of people from whom information and opinions are needed. In our experience, questionnaires are a common technique with systems intended for use outside the organization (e.g., by customers or vendors) or for systems with business users spread across many geographic locations. Most people automatically think of paper when they think of questionnaires, but today more questionnaires are being distributed in electronic form, either via e-mail or on the Web. Electronic distribution can save a significant amount of money as compared to distributing paper questionnaires.



Good Questionnaire Design

Selecting Participants As with interviews and JAD sessions, the first step is to identify the individuals to whom the questionnaire will be sent. However, it is not usual to select every person who could provide useful information. The standard approach is to select a *sample*, or subset, of people who are representative of an entire group. Sampling guidelines are dis- cussed in most statistics books, and most business schools include courses that cover the topic, so we do not discuss it here. The important point in selecting a sample, however, is to realize that not everyone who receives a questionnaire will actually complete it. On average, only 30 to 50 percent of paper and e-mail questionnaires are returned. Response rates for Web-based questionnaires tend to be significantly lower (often only 5 to 30 percent).

Designing a Questionnaire: Developing good questions is critical for questionnaires because the information on a questionnaire cannot be immediately clarified for a confused respondent. Questions on questionnaires must be very clearly written and leave little room for misunderstanding, so closed-ended questions tend to be most commonly used. Questions must clearly enable the analyst to separate facts from opinions. Opinion questions often ask the

respondent the extent to which they agree or disagree (e.g., Are network problems common?), whereas factual questions seek more precise values (e.g., How often does a network problem occur: once an hour, once a day, once a week?). For guidelines on questionnaire design.

Administering the Questionnaire The key issue in administering the questionnaire is get- ting participants to complete the questionnaire and send it back. Dozens of marketing research books have been written about ways to improve response rates. Commonly used techniques include clearly explaining why the questionnaire is being conducted and why the respondent has been selected; stating a date by which the questionnaire is to be returned; offering an inducement to complete the questionnaire (e.g., a free pen); and offering to supply a summary of the questionnaire responses. Systems analysts have additional techniques to improve response rates inside the organization, such as personally handing out the questionnaire and personally contacting those who have not returned them after a week or two, as well as request- ing the respondents' supervisors to administer the questionnaires in a group meeting.

Questionnaire Follow-up It is helpful to process the returned questionnaires and develop a questionnaire report soon after the questionnaire deadline. This ensures that the analysis process proceeds in a timely fashion and that respondent who requested copies of the results receive them promptly.

Document Analysis

Project teams often use document analysis to understand the as-is system. Under ideal circumstances, the project team that developed the existing system will have produced documentation, which was then updated by all subsequent projects. In this case, the project team can start by reviewing the documentation and examining the system itself.

Observation

Observation, the act of watching processes being performed, is a powerful tool for gathering information about the as-is system because it enables the analyst to see the reality of a situation, rather than listening to others describe it in interviews or JAD

	Interviews	Joint Application Design	Questionnaires	Document Analysis	Observation
Type of information	As-is, improvements, to-be	As-is, improvements, to-be	As-is, improvements	As-is	As-is
Depth of information	High	High	Medium	Low	Low
Breadth of information	Low	Medium	High	High	Low
Integration of information	Low	High	Low	Low	Low
User involvement	Medium	High	Low	Low	Low
Cost	Medium	Low–Medium	Low	Low	Low–Medium

Comparison of Requirements-gathering Techniques:

Summary and Conclusions

Given the global business environment, cultural issues become even more important when deploying information systems today. The cultural dimensions that need to be taken into consideration include power distance, uncertainty avoidance, individualism versus collectivism, masculinity versus femininity, and long- versus short-term orientation. Furthermore, each of these dimensions tends to interact with each other. From an information systems deployment perspective, the most important thing to remember is to take into consideration the local culture before deploying the new system.