Introduction to Systems Analysis and Design

What is a System?

A system is a set of interrelated components that function together to achieve a common goal. The components of a system are called subsystems.

The components of a system are interdependent; that is, the output of one subsystem is usually becomes the input of another subsystem. Thus, malfunctioning of one component affects the functioning of other components.

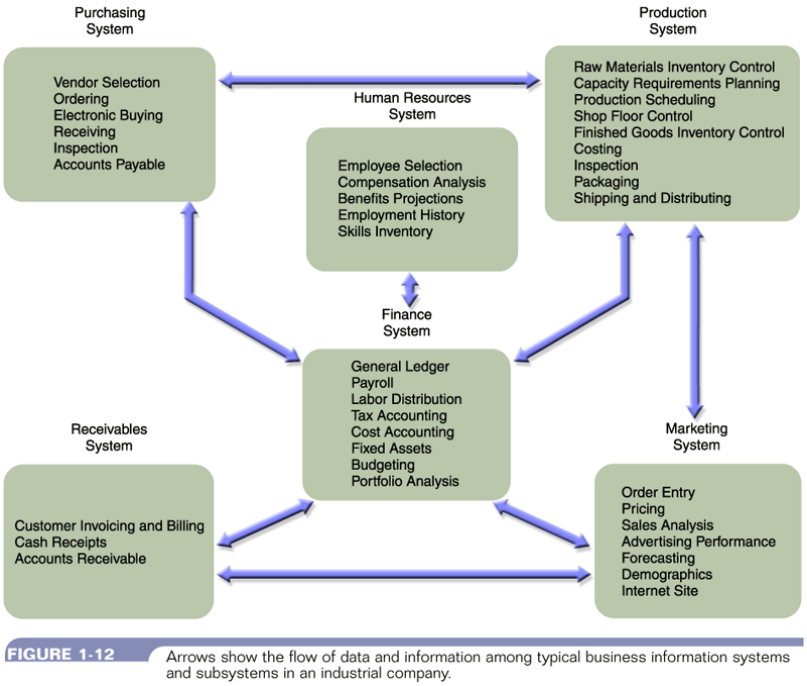


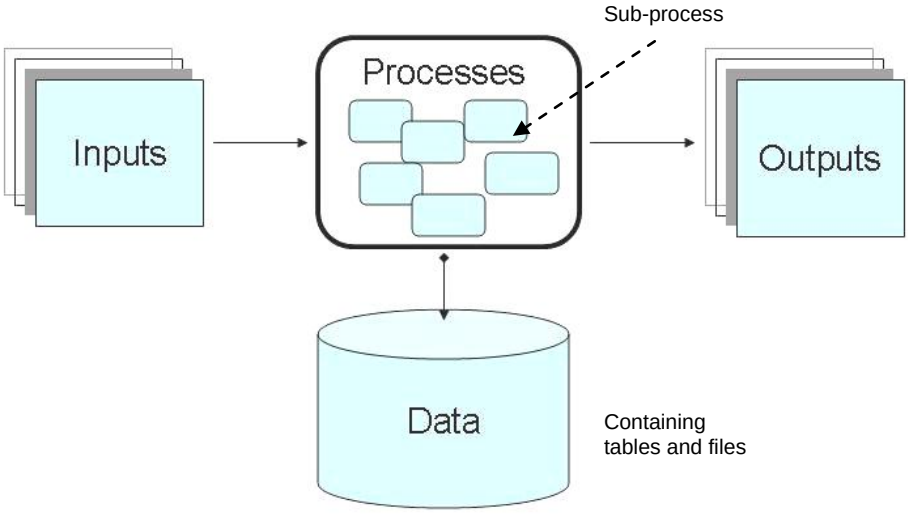
FIGURE 1-12 Arrows show the flow of data and information among typical business information systems and subsystems in an industrial company.

A system receives inputs from the outside environment, which are then processed by various subsystems, and then delivers required outputs to the outside environment.

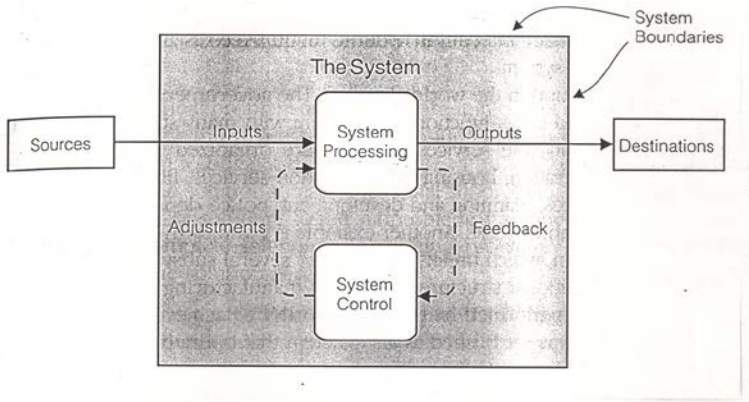
A system also has control mechanisms to make certain decisions. This is usually performed as a feedback to the system user (or automated to the system environment) followed by certain decisions.

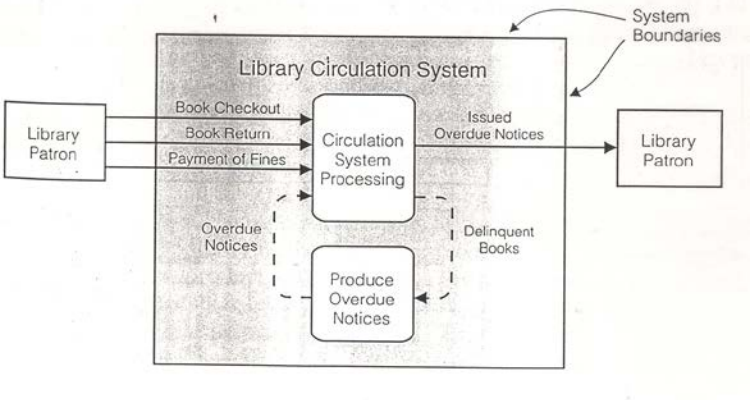
- An automated system control might be a heating and cooling thermostat which turns on and off at a presetting temperature.

    - A library circulation system is another example of system control, which may require user interaction. When books are overdue, the system alerts the user. The user then has the option of printing an overdue notice. This process can also be automatic.



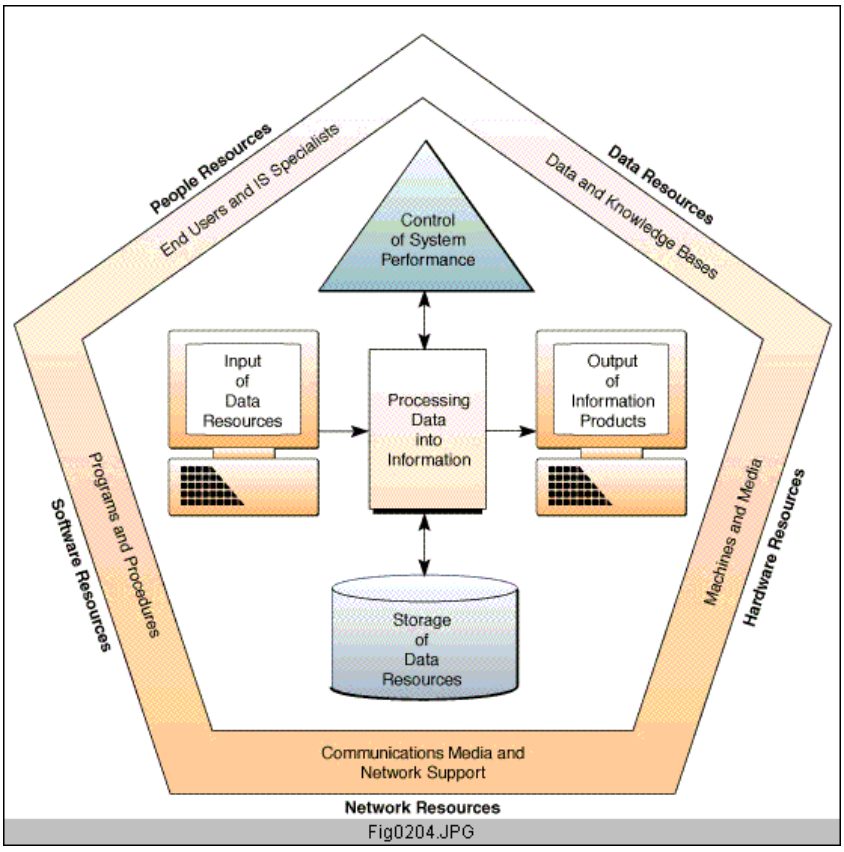
The concept of a system including inputs, outputs, processes, and data storage. You also need networks to accomplish the system tasks.





An organization deals with various types of information systems, such as, transaction processing systems, management information systems, decision support systems, and expert systems. Each of these systems requires many subsystems to accomplish its goals.

A computer-based information system deals with the organizational data, computer hardware, software, network, people, and procedures.



Systems Analysis and Design

Systems analysis and design is a complex, challenging, and stimulating **organizational process** that a team of business and system professionals uses to develop and maintain computer-based information systems.

Systems are built and rebuilt in an organization to increase value to the business of an organization. Thus, system analysis and design is a process that is used to analyze, design, and implement improvements in the functioning of business that can be accomplished through the use of computerized information systems.

System analysis and design uses a methodological approach to develop a computer-based information system.

The Systems Development Life Cycle

The systems development life cycle (SDLC) is a process by which systems analysts, software engineers, and programmers build systems.

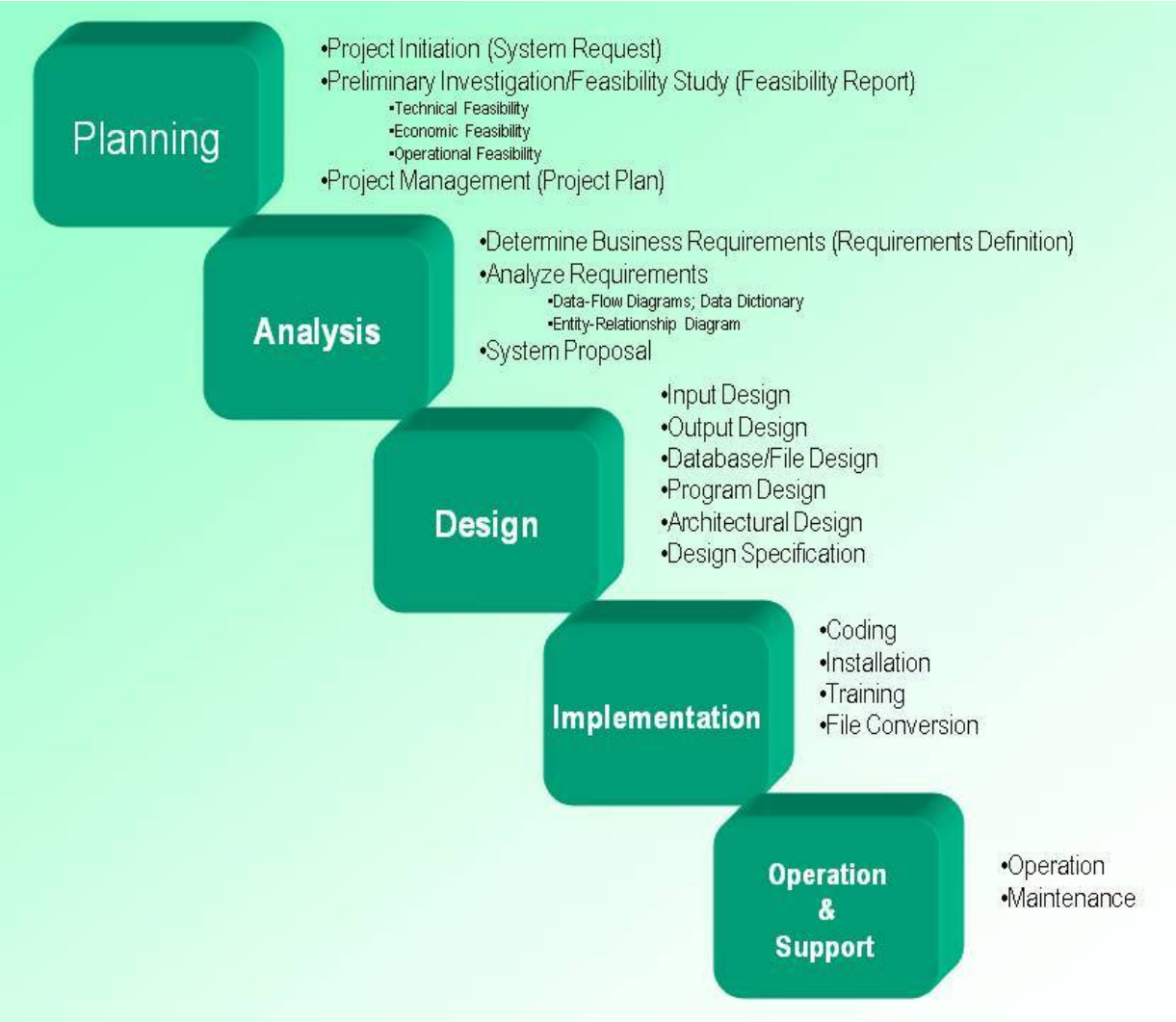
It is a phased approach to analysis and design of a system, which holds that systems are best developed through the use of a specific cycle of analyst and user activities.

In its simplest form, SDLC consists of five phases:

- systems planning
- systems analysis
- systems design
- systems implementation
- systems operation and support

Each phase is divided into multiple steps or activities that need to be completed.

Note a system analyst spends most of his or her time in analyzing and designing a system before the programmers actually develop the system; although program coding takes about three-fourth of the total time of the system development life cycle.



Typical Phases and Activities of a Systems Development Life Cycle

Systems Development Methodology

A methodology is a formalized approach of developing a business system. SDLC provides the guidelines for the steps or activities that need to be performed to design and develop a systems; it does not prescribe the method of performing the phases or activities.

A methodology provides a sequential approach of traversing the activities or tasks of designing and developing a system. There can be multiple methodologies of developing a system; however, there is only one systems development life cycle.

Example: You wish to fly from one place to another; however, you can go through various routes to reach your destination. Going from one location to another is SDLC, and taking various routes are methodologies.

The following is an example of the Waterfall development methodology:

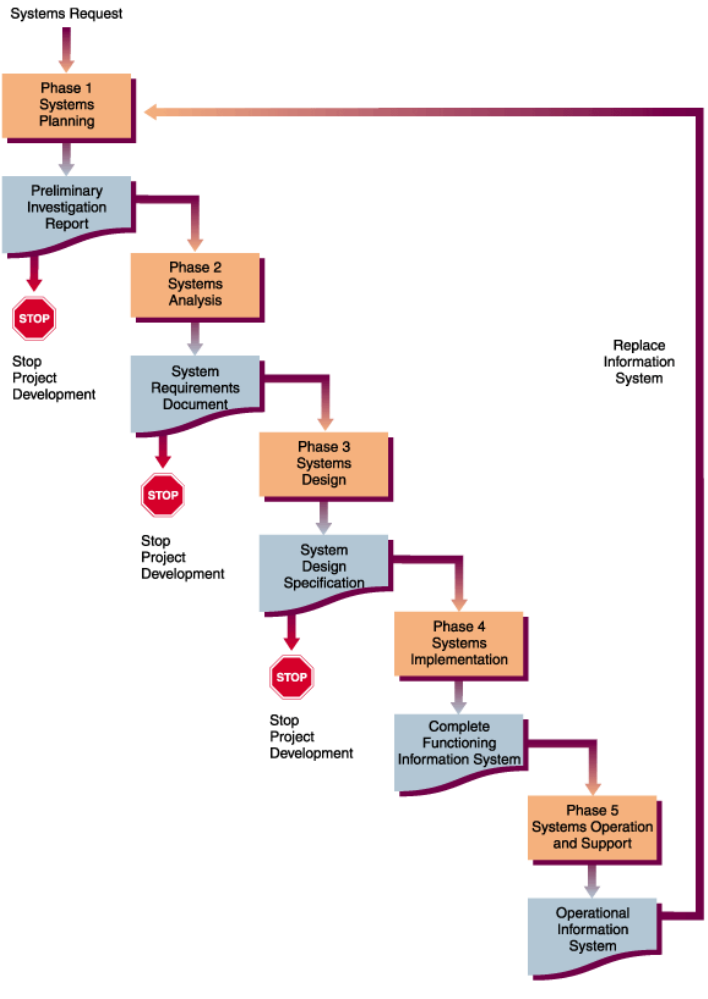


FIGURE 1-24 The phases and end products of the systems development life cycle (SDLC).

System Development Methodologies

A methodology is a formalized approach to develop a business system. There are several methodologies that are used to develop a system. All of these follow certain phases of the system development life cycle. Considering the **time** requirement to develop a system, these methodologies can be grouped into the following:

- Structured Design
- Rapid Application Development (RAD)

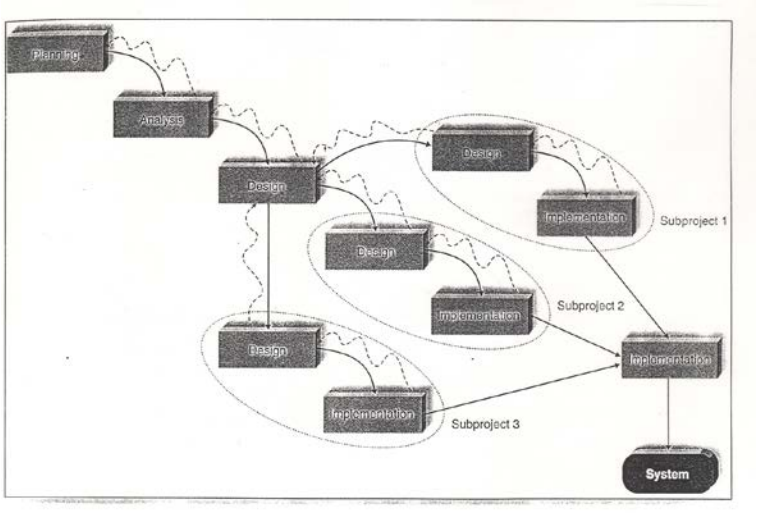
Structured Design

Structured design methodologies adopt a formal step-by-step approach to the SDLC that moves logically from one phase to the next. Structured design methodologies can be divided into several types as described in the following:

Waterfall Development: This is the original structured design methodology mentioned above. In this case, each phase of the SDLC flows downward in a sequence like a waterfall. The key deliverables for each phase are typically produced on paper (often as long documents) and are presented to the project sponsor for approval.

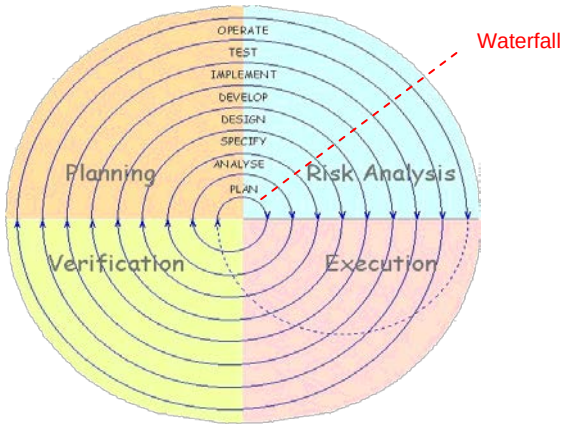
Although it is possible to go backward in the SDLC (e.g., from design to analysis), it requires substantial resources. Thus the activities of each phase are completed before moving to the next phase.

Parallel Development: The parallel development methodology attempts to address the problem of long delays between the analysis phase and the delivery of the system. Instead of doing the design and analysis in sequence, it performs a general design of the whole system and then divides the project into a series of distinct subprojects that can be designed and implemented in parallel.



***Spiral Model*:** The "spiral model" is an iterative model for software development, described by Barry Boehm. Each iteration basically follows the Waterfall Model, but subsequent iterations build upon previous work. Barry Boehm devised the spiral model to address the weaknesses of the waterfall model, especially its lack of resilience in the face of change. The spiral model focuses on addressing *risks* incrementally by repeating the waterfall model in a series of *cycles* or *rounds*:

- Concept of Operation
- Software Requirements
- Software Product Design
- Detailed Design
- Code
- Unit Test
- Integration and Test
- Acceptance Test
- Implementation



It is called a "spiral" because the common illustrative diagram shows a long string of waterfalls wrapping around a center, emphasizing the aspect of continuous growth and revisiting of previous work. The spiral model is an improvement on the waterfall model, as it provides for multiple builds and provides several opportunities for customer involvement. However, it is elaborate, difficult to manage, and does not keep all workers occupied during all phases.

The spiral model is an approach to application development that focuses on minimizing risk. Each transition around the spiral involves repeating the same four steps: (1) determine the objectives, alternatives, and constraints of this iteration; (2) evaluate alternatives; identify and resolve risks; (3) develop and verify deliverables from this iteration; and (4) plan the next iteration.

Iterative Model: The activities of phases for planning, analysis and design are repeated until a solid system design is developed before developing the system.

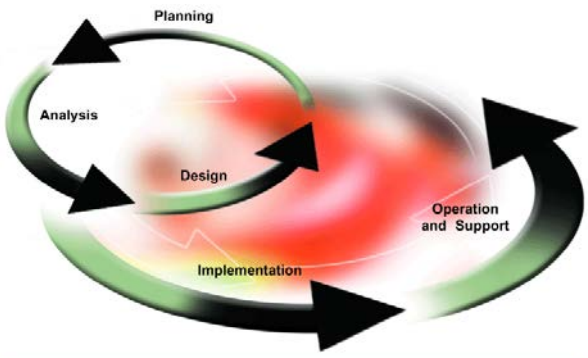


FIGURE 1-25 An alternative model of the SDLC that shows the interaction of planning, analysis, and design, which leads to implementation and then to operation and support.

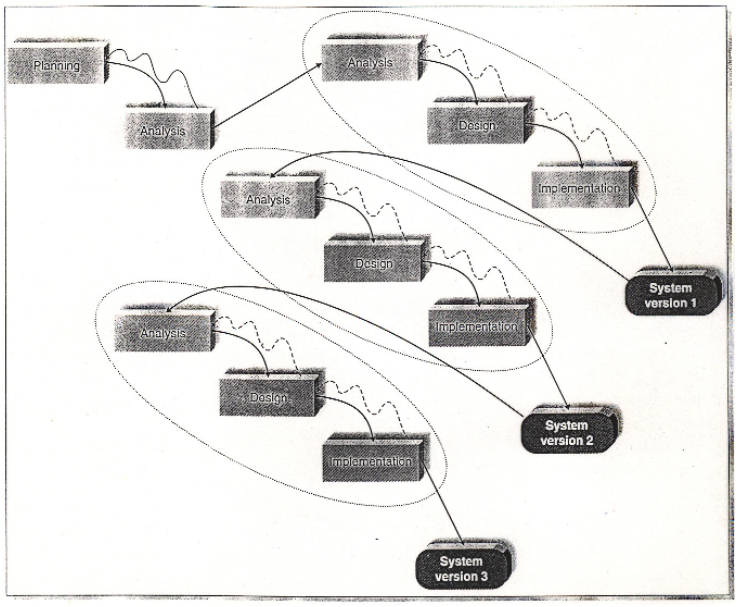
Rapid Application Development (RAD)

Rapid application development attempts to address both weaknesses of the structured development methodologies: *long development times* and difficulty in understanding a system from a *paper-based description*.

RAD methodologies adjust the SDLC phases to get some part of the system developed quickly and place into the hands of the users, so that users can better understand the system and provide feedback for improvement of the system.

Two common methodologies of rapid application development are: *phased development* and *prototyping.*

> ***Phased Development:*** The phased development methodology breaks the overall system into a series of versions that are developed sequentially. The analysis phase identifies the overall system concept, and the project team, users, and system sponsors 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.
>
> Once version 1 is implemented, work begins on version 2 and follows the steps, and so on. Any additional requirement identified during testing of the older version is implemented in the next version.
>
> Phased development has the advantage of quickly getting a useful system into the hands of the users.

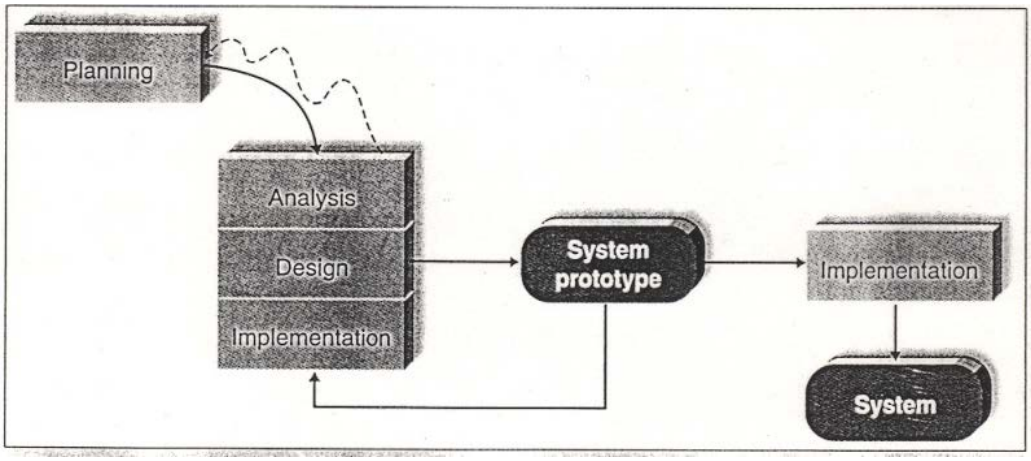


Prototyping: The prototyping methodology performs the analysis, design, and implementation phases concurrently, and all three phases are performed repeatedly in a cycle until the system is completed.

In this case, the users of the system are an active participant of the system development process. With the basic requirements from the users, a quick analysis and design of the system is performed, and a prototype of the system containing main features of the requirements, are developed.

The prototype is handed to the users for testing and to provide comments; which are then reanalyzed and redesigned, and a second prototype is developed. The process continues in a cycle until the users and developers agree to a final system.

Prototyping methodology gives a system quickly to the users' hands, which they can visualize from the very beginning instead of waiting at the end of the system development.



Tools for Rapid Applications Development:

Most RAD methodologies recommend that analysts use special techniques and computer tools to speed up the analysis, design, and implementation phases. Two common techniques and tools include JAD (joint application design), and CASE (computer-aided software engineering) tools.

Joint Application Design (JAD): In this approach, the sponsor company creates a task force of users, managers, and IS professionals that works together to gather information, discuss business needs, and define the new *system requirements*. This group usually meets over periods of days or weeks.

Because of the wide range of user input, JAD produces the best possible *definition of a new system* than a single analyst can provide.

CASE (Computer-Aided Software Engineering)Tools: There are software packages available that can help speed up various phases of the SDLC, thus shortening the development time. These tools can be used in the *analysis phase* to document data dictionary (data elements, data flows, data stores, processes, and external entities), and create data-flow diagrams.

In the *design phase*, CASE tools can be used to create entity-relationship diagrams and to generate program codes in certain language such as Visual Basic.

Visible Analyst, a CASE tool from the *Visible Systems Corporation* can be used in the planning, analysis, design, and code generation of a system development.

An Overview of the Systems Development Life Cycle

As it was mentioned before, in the simplest form, SDLC consists of five phases:

- systems planning (preliminary investigation)
- systems analysis
- systems design (logical and physical design)
- systems implementation (programming and installation)
- systems maintenance (operation and support).

The number and names of the SDLC phases varies from author to author, but the analysis, design and implementation phases are common. Some authors break the planning, design, and implementation phases further. In some cases, systems maintenance is not strictly considered as part of the SDLC. But in all cases, the following issues are addressed in developing a system:

- Identify a project
- Determine end-user's requirements
- Analyze system needs
- Acquire computer hardware and software (if required)
- Design the new system
- Construct the new system
- Install the new system
- Maintain and improve the system

In each phase of the SDLC, certain steps are performed to complete the phase. In a particular project, all steps in a particular phase may not follow a logical path, but the steps are fitted according to the need of the system.

An important outcome of each phase of the SDLC is a document. The document from one phase provides a structure for the development of the next phase. At each phase the system gets more elaborate and refined.

As the name says, the *systems analysis and design* course is mainly designed to cover the methodologies used in the *analysis* and *design* phases of a systems development life cycle.

Systems Planning

The systems planning phase is the fundamental process of understanding why an information system should be built and determine how the project team will go about building it. Thus the planning phase can be divided into two sub-phases: *project identification and selection, and project initiation and planning.*

Project Identification and Selection:

In this phase, someone in an organization identifies that there is a need to improve an existing system or a new system is needed to improve business operations. Most ideas come from outside the IT department such as marketing, accounting, etc., and managers and other personnel get involved to initiate a process.

The initiation of this phase comes as a ***system request*** document, which provides a brief summary of a business need and explains how a new or improved system can increase business value of the organization.

Project Initiation and Planning:

In this phase, an analyst from the IT department gets involved. His or her purpose is to identify clearly the nature and scope of the problems, opportunities of improvement, and define specific solutions. This requires *preliminary investigation*.

Activities in this phase include interviewing prospecting users and management, summarizing the knowledge obtained, estimating the scope of the project, and documenting the results.

The preliminary investigation is often called a *feasibility study (or feasibility analysis)* as the information gathered by the analyst are analyzed in terms of *economical* feasibility, *technical* feasibility, and *operational* feasibility of the project:

- *Technical Feasibility*: Is the solution technically practical? Does our staff have the technical expertise to design and build this solution?

- *Operational Feasibility*: Will the solution fulfill the end-user's requirements? To what degree? How will the solution change the end-user'' work environment? How do end-users feel about such a solution?

- *Economic Feasibility*: Is the solution cost effective?

The output of the planning phase is a ***feasibility report*** that presents findings and recommendations. It usually includes a problem definition, objective summary, and preliminary cost-benefit analysis. The objective of this report is to determine whether the project is feasible, i.e., whether to invest significant resources for further study or not.

The system request and the feasibility report are presented to an information systems *approval committee*. If the committee approves the project, then the second step of the project initiation occurs – *project management*.

During project management, the *project manager* creates a *work plan,* staffs the project, and puts techniques in place to help him or her control and direct the project through the entire SDLC.

The deliverable for project management is a ***project plan*** that describes the project team member roles, deliverable phases, cost of each deliverable, timeline for each deliverable, and so on.

Having a project plan at the beginning keeps the project under control in terms of time and money. It also helps to stop the project at any step if the project runs over the budget or the business policy of the sponsor-company changes over time.

Systems Analysis

In this phase, the system analyst tries to understand the data and information collected in the previous phase and define requirements or needs of the new system.

The systems analyst may collect further information from the users to obtain a clear picture of the existing system and understand the requirements of the new system. He or she then analyzes the requirements - an elaborate step termed as *requirement analysis*.

The most popular approach to requirement analysis is *modeling*. The analyst may use *process modeling tool* (data flow diagrams) to chart the input, processing, and output of the business functions in a structured graphical form. The analyst also uses *data modeling* tool such as Entity-Relationship (E-R) diagrams.

The analysis leads to the *data dictionary* (definitions of the inputs, files, database, and outputs) and *process descriptions* for the new system, but do so without expressing technical details. Essentially, the activities in this phase focus on the general or logical design of the system (as opposed to physical design) to give an overview of the system.

An alternative approach to requirement definition is *prototyping*. When end-users have difficulty in defining requirements, the analyst uses computer tools to build a prototype, or small-scale working model of the final system. The end-users can then react to the prototype to help the analyst establish their requirements more easily.

During this phase, the system analyst analyzes various possible solutions to the problem. Each candidate is evaluated for technical, operational, and economic feasibilities. Among the possible candidate solutions analyzed, a solution is selected that defines the objectives of the project. The selection phase determines how the new system is to be designed but only at a very high level - no details!

The solution that offers the best combination of technical, operational, and economic feasibility may be selected for the next phase of the SDLC. The output of this phase is a ***system proposal (or systems requirement document).***

The system proposal is presented to the *approval committee.* If the committee approves the project, a detail design of the system is initiated.

Systems 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 that will be used; and the specific programs, databases, and files that will be needed. Look and feel or blueprint of the system on paper is developed in this phase.

Systems design phase can be broken into two phases: *logical design and physical design.*

Logical Design:

The logical design addresses the business aspect of the system, which is independent of any hardware. In this phase, the system analyst uses the information collected earlier to develop a logical design of the information system. It includes the following:

- *Database structure:* Identifies external entities and relationships through E-R diagrams and normalization. Also mentioned in the analysis phase.

Physical Design:

The logical design is implemented through the physical design. It identifies the file and database structures, system structure, program structure, and hardware and software necessary to implement the system. They include:

- *Database:* Implementation of E-R diagram. Defines all tables in the database including field names, field size, data type, keys, validation rules, and so on.

- *Files:* If the system requires any input or output file, their structures are defined. A file definition includes information such as the record length, field names, field size, field sequence, and so on.

- *Data-Entry Forms:* Data-entry forms are developed, which defines the relationships of data-entry fields with the file or table-fields. The forms can also be used for displaying data. These forms can be character-based or GUI-based. The analyst defines exactly how each form looks on the computer screen.

- *Menus:* Menus are designed such that the forms and reports can be accessed and printed according to the business need of the organization. It should also address the security requirement of the organization.

- *Reports:* Reports are defined, which includes the relationships of the report-fields with the table-fields, any calculations, layout, font type, font size, and so on.

- *Systems and Programs:* All programs and program modules corresponding to processes are defined through a structured chart. To define the processes

performed by a program, the analyst may include data-flow diagrams, flow-charts, and pseudocodes that were used to describe the processes in the analysis phase.

- *Programming Languages:* Identifies the languages that will be used to code the systems.
- *Database System:* Identifies the database software that will be used in the system.
- *Hardware Platform:* Identifies the computer hardware that will be used in various parts of the system.
- *Operating Systems:* Operating systems that will be used in various parts of the system.
- *Network Architecture:* Identifies the network architecture of the system.

The output of the design phase is **a *Functional Specification or Design Specification***. Programmers and network administrators use this document as the working specification for the physical development of the system.

Systems Implementation

This phase can be divided into two parts: *systems development* and *systems installation*.

Systems Development:

In this phase, the system analyst works with the programmers to develop the new system (software and hardware) according to the definition of the functional specification. The analyst works as a facilitator to the programmers and other system architects to explain any part of the design that may not be clear from the document.

Constructing the new system is the most time-consuming part of the development life cycle. If the functional specification defines all processes clearly, then the programming becomes a systematic activity. A poor functional specification may require a lot of interaction time between the analyst and programmers.

The development phase also involves program testing and control. As each input form, report or process is developed, it is tested and kept aside until the whole system is integrated and tested again.

During this phase, the analyst also works with the users to develop effective documents for software. These might include procedure manual, online help, frequently asked questions, “readme” files, training guide, and so on.

Systems Installation:

Once the system is developed, the analyst helps implement the new system. In this phase, the analyst works with the users to describe the functionality of the new system. He or she also trains the users for the new system.

It may take several versions to install the final system. Any change or improvement originated due to user interaction or demonstration should be reflected in the next version, until the users are satisfied with the system.

Systems Operation and Maintenance

Once a system is delivered, the analyst's role changes from development to support.

Maintenance is the correction of errors and omissions that were not caught during the testing and delivery phases. Improvements are the additions of new capabilities to the system. They may arise due to the reasons such as:

- *Additional features:* Request for additional features such as new reports and interfaces, improved screen or report layouts, and so on.

- *Change of business over time:* Software must be modified to encompass changes as new government rules need to be implemented.

- *Hardware and software change:* A system that uses older technology may be modified to use the capabilities of newer technology.

The phases and steps defined in the SDLC will be discussed in detail throughout the semester.

Information Systems Department

Most organizations now have an information systems department that is responsible for the development and maintenance of information systems. Various personnel are involved in the IS department.

To develop an information system, a project team is selected containing a project manager, business analyst, systems analyst, network administrator, and programmers.

The project manager is responsible for ensuring that the project is completed on time and within budget. His or her role includes managing the team members, developing the project plan, assigning resources, and being the primary point of contact to the project sponsor. The project manager must have significant experience as a systems analyst and project management.

The systems analyst is by far the most important person in the overall success or failure of the project development.

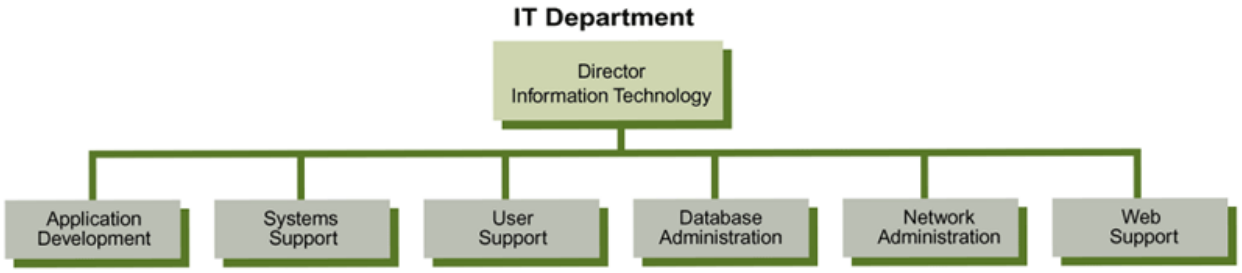


FIGURE 1-26 This organization chart shows the major functions of the IT department staff.

The Role of a System Analyst

A system analyst studies the problems and need of an organization to determine how people, methods, and computer technology can best accomplish improvements for the business.

When computer technology is used, the analyst is responsible for the efficient capture of data from its business source, the flow of that data to the computer, the processing and storage of that data by the computer, and flow of useful and timely information back to business users.

A systems analyst should have the following background and knowledge:

- *Knowledge of SDLC:* Essentially, the systems analyst performs systems analysis, systems design, systems implementation, and systems support for computer-based business applications.

- *Technical Knowledge:* An analyst does more than analyze and design systems. The analyst must be experienced in working with computers and must exploit the latest technological innovations.

- *Programming Knowledge:* He or she must have enough computer experience to program, to understand the capabilities of computers, to glean information requirements from users, and to communicate what is needed to programmers.

- *Problem Solver:* The analyst is a problem solver. He or she must have the capability to take a large business problem, break that problem down into its component parts, analyze various aspects of the problem, and then assemble a system to solve the problem. Analysts must be able to creatively define alternative solutions to problems and needs.

- *Knowledge of Tools and Techniques:* The analyst plays many roles, sometimes balancing several at the same time. He or she is a person who views the analysis of problems as a challenge and who enjoys devising workable solutions. When necessary, the analyst must be able to systematically tackle the situation at hand through skillful application of tools, techniques, and experience.

- *Good Communicator:* The analyst must be a good communicator – capable of relating meaningfully to other people over extended periods of time. He or she must be able to work with people of all descriptions. The analyst must be willing and able to deal with his or her business clients (end-users), business management, programmers, information system management, auditors, and information system salespeople.

- *Teamwork:* System analysts work in teams, and so the ability to function as part of a term, to cooperate and compromise, is critical for the success of most projects.

- *Business Knowledge:* The system analyst need to have some general business knowledge, so that he or she can communicate with business experts to gain knowledge of problems and needs.

In many organizations, a systems analyst's role may not be well defined. A computer programmer may perform some or all of the responsibilities of an analyst. In some organization, a system analyst works as a programmer.

The most common title is programmer/analyst, whose job includes the responsibilities of both the computer programmer and the systems analyst. Other titles include systems engineer, systems designer, operations specialist, information specialist, data analyst, and business analyst.
