System Analysis and Design (CACS203)

Session : 1st

Course Contents

Unit 1 System Development Fundamentals **9 Hrs.**

a. The Systems Development Environment

Introduction, Modern Approach of System Analysis and Design, Information System and its Type, Developing Information Systems and the Systems Development Life Cycle, The Heart of the Systems Development Process, The Traditional Waterfall SDLC, Approaches for Improving Development, CASE Tools, Rapid Application Development, Service-Oriented Architecture, Agile Methodologies, eXtreme Programming, Object- Oriented Analysis and Design

b. The Origins of Software

Introduction, System Acquisition, Reuse

c. Managing the Information Systems Project

Introduction, Managing Information Systems Project, Representing and Scheduling Project Plans, Using Project Management Software

Unit 2 Planning **7 Hrs.**

a. System Development Projects: Identification and Selection

Introduction, Identifying and Selecting Systems Development Projects, Corporate and Information Systems Planning

b. System Development Projects: Initiation and Planning

Introduction, Initiating and Planning Systems Development Projects, Process of Initiating and Planning IS Development Projects, Assessing Project Feasibility, Building and Reviewing the Baseline Project Plan

Unit 3 Analysis **13 Hrs.**

a. System Requirements

Introduction, Performing Requirements Determination, Traditional Methods for Determining Requirements, Contemporary Methods for Determining System Requirements, Radical Methods for Determining System Requirements,

Requirements Management Tools, Requirements Determination Using Agile Methodologies

b. System Process Requirements

Introduction, Process Modeling, Data Flow Diagramming Mechanics, Using Data Flow Diagramming in the Analysis Process, Modeling Logic with Decision Tables

c. System Data Requirements

Introduction, Conceptual Data Modeling, Gathering Information for Conceptual Data Modeling, Introduction to E-R Modeling, Conceptual Data Modeling and the E-R Model, Representing Super-types and Sub-types, Business Rules, Role of Packaged Conceptual Data Models – Database Patterns

Unit 4 Design **12 Hrs.**

a. Designing Databases

Introduction, Database Design, Relational Database Model, Normalization, Transforming E-R Diagrams into Relations, Merging Relations, Physical File and Database Design, Designing Fields, Designing Physical Tables

b. Designing Forms and Reports

Introduction, Designing Forms and Reports, Formatting Forms and Reports, Assessing Usability

c. Designing Interfaces and Dialogues

Introduction, Designing Interfaces and Dialogues, Interaction Methods and Devices, Designing Interfaces and Dialogues in Graphical Environments

Unit 5 Implementation and Maintenance **4 Hrs.**

a. System Implementation

Introduction, System Implementation, Software Application Testing, Installation, Documenting the System, Training and Supporting Users, Organizational Issues in Systems Implementation

b. System Maintenance

Introduction, Maintaining Information Systems, Conducting Systems Maintenance

The Systems Development Environment

Introduction

- System: A system is a group of elements or components which work together to accomplish a common Task
- **Characteristics of system:**

1. Predetermined Objectives
2. Organization
3. It can be further subdivided
4. All components can be interdependent or interrelated

Information system

- Companies uses information as a weapon in the battle to increase productivity, deliver quality products and services and make sound decisions.

- Information technology is combination of hardware and software products and services.

Why do we need system development?

- To create new one
- Updating existing one.

The Systems Development Environment

Introduction

Information Systems Analysis and Design (ISAD)

- Complex organizational process.
- Used to develop and maintain computer-based information systems.
- Used by a team of business and systems professionals.
- An organizational improvement process (respond to and anticipate problems).
- Uses technology (Internet, WWW marketing, online business, eBay, Amazon.com etc).

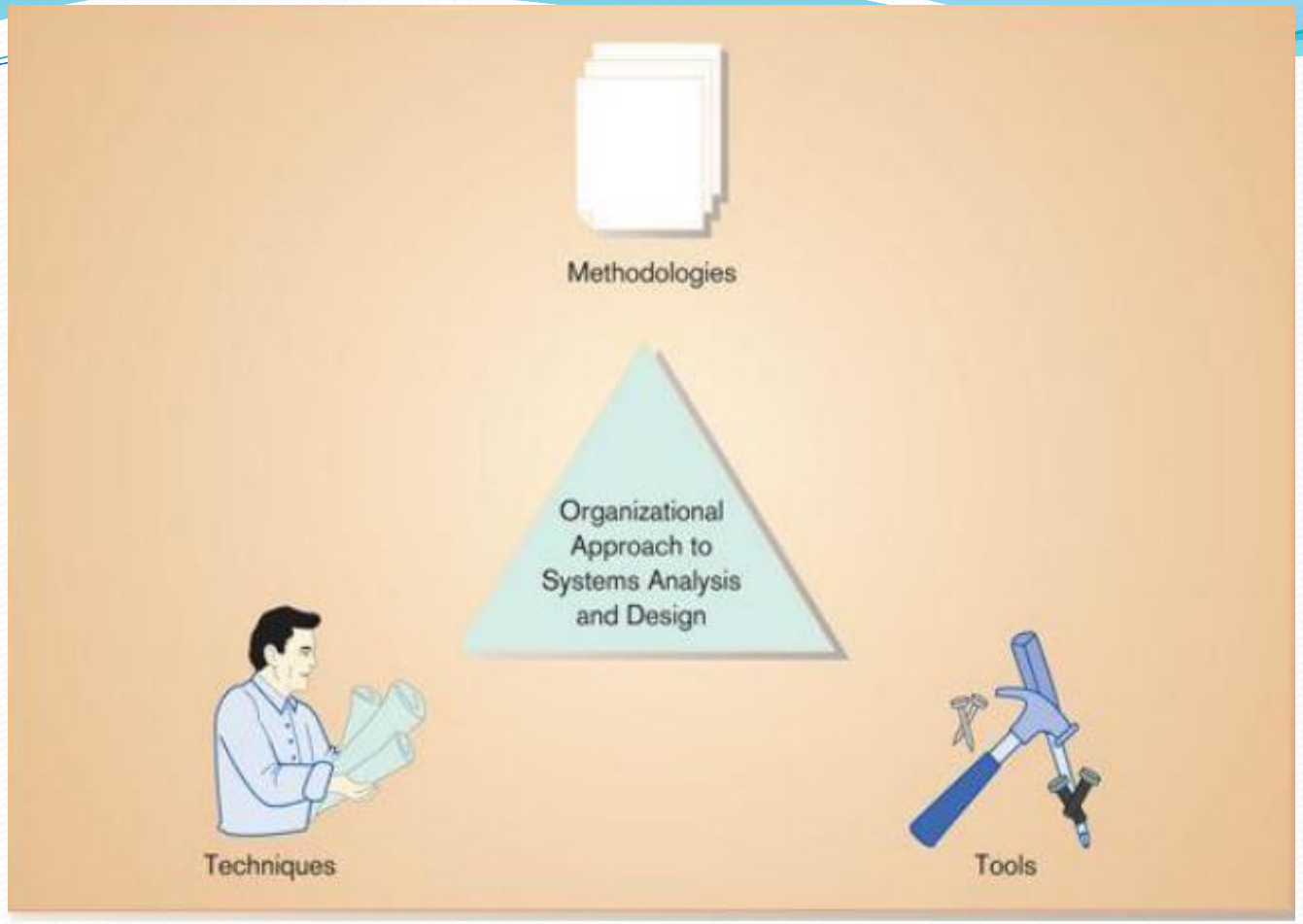


Figure 1-1 An organizational approach to systems analysis and design is driven by methodologies, techniques, and tools

- An output of system analysis and design is **application software.**
- **Methodologies** are comprehensive, multiple-step approaches to system development. Methodologies uses different techniques.
- **Techniques** are particular processes that you as an analyst, will follow to ensure your work is well thought out, complete, and comprehensive to others on your project team. Eg: conducting interviews to determine what your system should do, diagramming the system's logic, designing the reports your system will generate.
- **Tools** are computer programs that make it easy to use and benefit from techniques.

Modern Approach to Systems Analysis and Design

- 1950s: All applications had to be developed in machine language or assembly language. They had to be developed from scratch because due to the absence of software industry
- 1960s: Smaller, faster, less expensive computers, beginning of the software industry, use in-house development.
- 1970s: Realized how expensive to develop customized information system for every application , started development of database management system.
- 1980s: , The software industry expended greatly, CASE(computer aided software engineering) tools.

- Started writing application software in oop languages, graphics were used, developed less software in-house and bought more from software vendors.
- 1990s: focus on system integration, GUI(Graphical user interface) applications, client/server platforms, Internet.
- The new century: Web application development, wireless PDAs (personal digital assistants, eg pocket PCs), ASP(application service provider).

Systems Analyst:

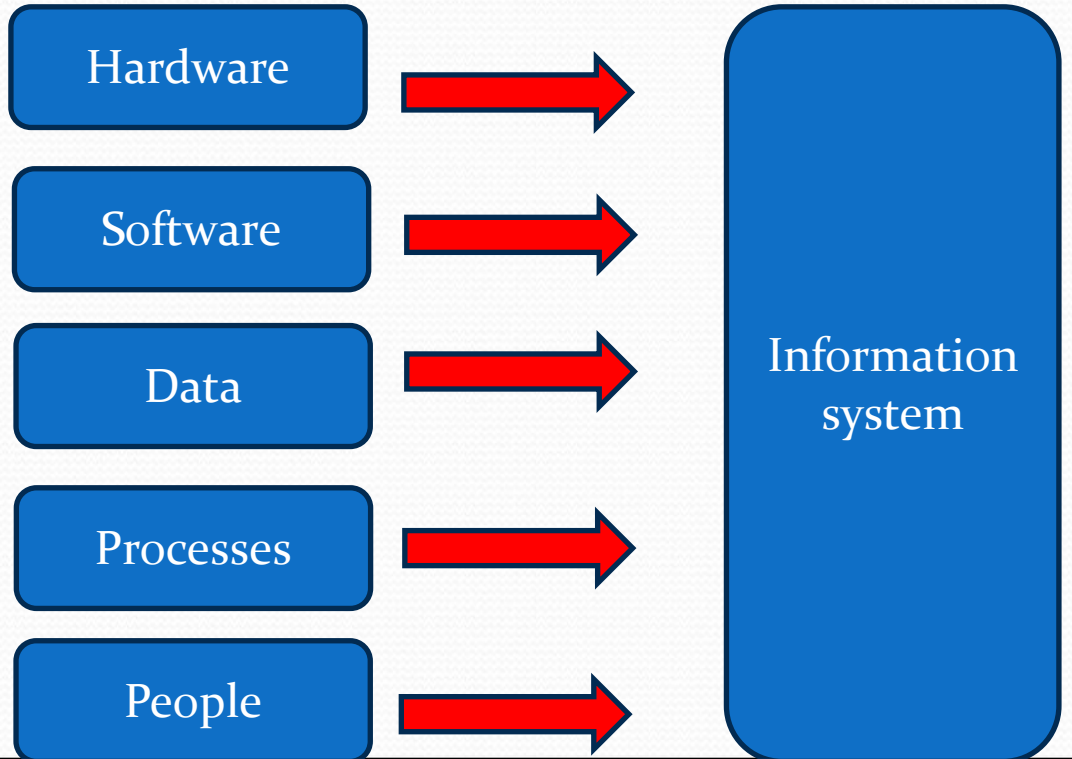
- Plan, develop and maintain information system
- Manages IT projects, including task, resources, schedule and costs.
- Conducts meetings, deliver presentation, writes report and documentation.

Roles of System Analyst:

- Defining IT requirements of organization
- Gathering Data/Facts
- Analyzing the problem
- Setting priority amongst requirements
- Problem solving
- Drawing Specification
- Designing System
- Evaluating System

What is an Information System?

- Information system is a collection of hardware, software, infrastructure and trained personnel which are going to do easy planning to make reliable infrastructure, control, coordination between software and hardware and decision making in an organization



Types of Information Systems

- Transaction Processing Systems (TPS)

✓ Transaction processing systems are used to record day to day business transactions of the organization. They are used by users at the operational management level.

✓ Automate handling of data about business activities (transactions)

✓ Process orientation

✓ Example: Point of sales System, Payroll System etc.

- Management Information Systems (MIS)

✓ Converts raw data from transaction processing system into meaningful form

✓ used by tactical managers to monitor the organization's current performance status.

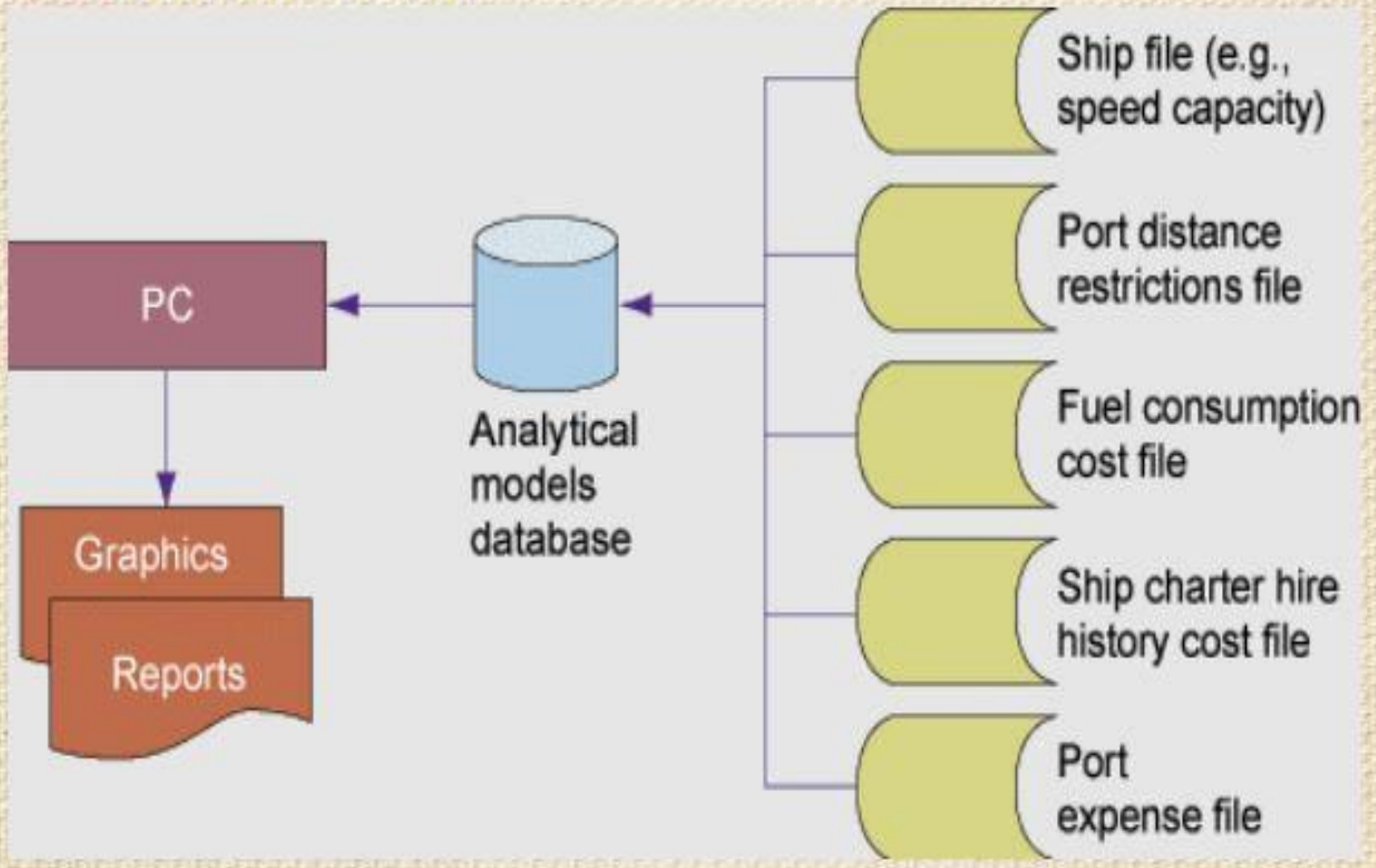
✓ Data orientation

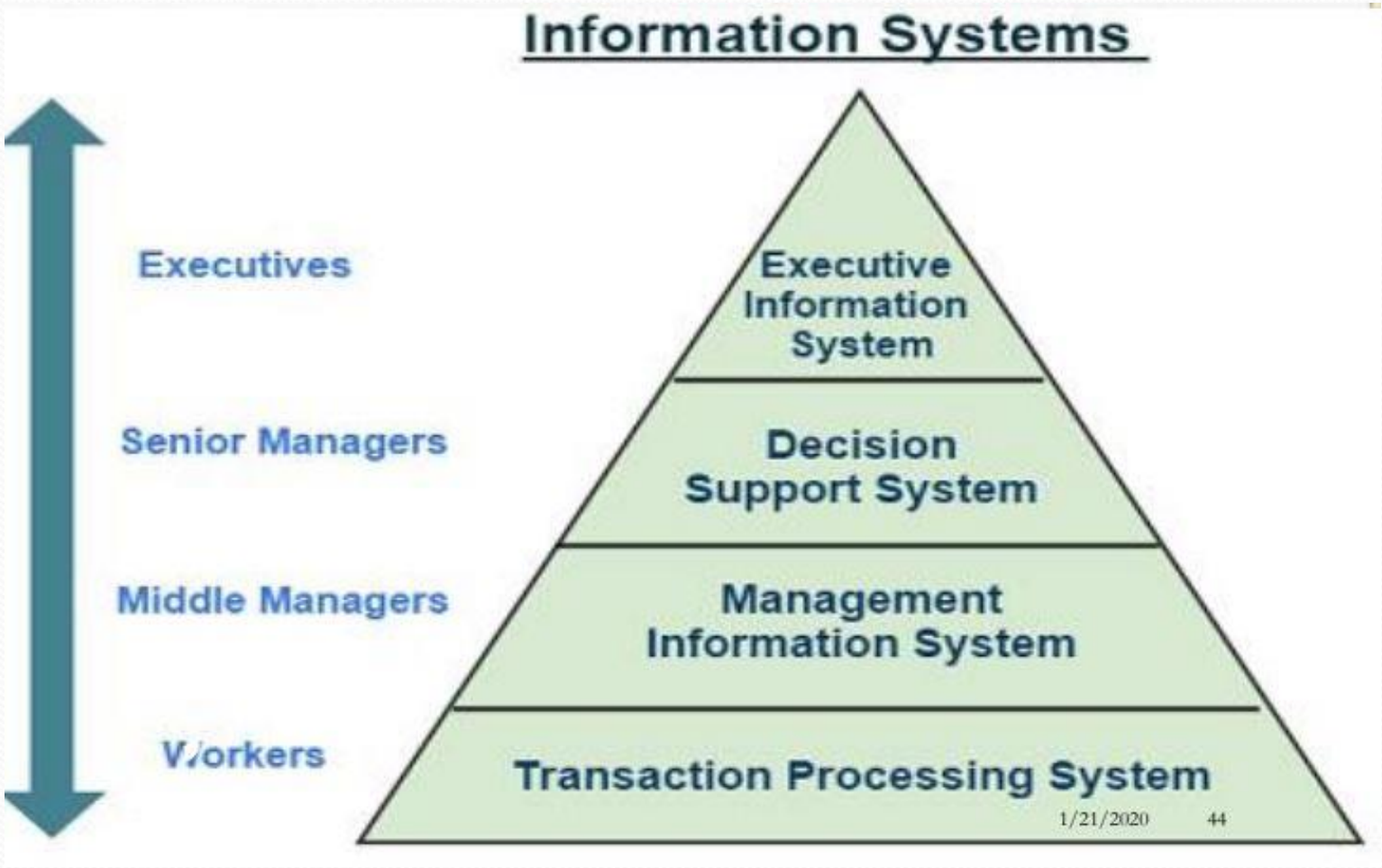
✓ Example: HRMS,Sales Management System

Decision Support Systems (DSS)

- Decision support systems are used by senior management to make non-routine decisions. Decision support systems use input from internal systems (transaction processing systems and management information systems) and external systems.
- Designed to help decision makers
- Provides interactive environment for decision making
- Involves data warehouses, executive information systems (EIS)
- Database, model base, user dialogue
- Example: Financial Planning System, Bank Loan Management System

Information systems at the ***management level*** of an organization that ***combine data and sophisticated analytical models to support*** *non-routine* ***decision making***




Information Systems
Executives
Senior Managers
Middle Managers
Workers
Executive Information System
Decision Support System
Management Information System
Transaction Processing System

Pyramid Diagram of Organizational levels and information requirements

Pyramid Diagram

Summary of Information Systems Types

Table 1-1 Systems Development for Different IS Types

IS Type	*IS Characteristics*	*Systems Development Methods*
Transaction processing system	High-volume, data capture focus; goal is efficiency of data movement and processing and interfacing different TPSs	Process orientation; concern with capturing, validating, and storing data and with moving data between each required step
Management information system	Draws on diverse yet predictable data resources to aggregate and summarize data; may involve forecasting future data from historical trends and business knowledge	Data orientation; concern with understanding relationships among data so data can be accessed and summarized in a variety of ways; builds a model of data that supports a variety of uses
Decision support system	Provides guidance in identifying problems, finding and evaluating alternative solutions, and selecting or comparing alternatives; potentially involves groups of decision makers; often involves semi-structured problems and the need to access data at different levels of detail	Data and decision logic orientations; design of user dialogue; group communication may also be key, and access to unpredictable data may be necessary; nature of systems requires iterative development and almost constant updating

Developing Information Systems

- System Development Methodology is a standard process followed in an organization to conduct all the steps necessary to analyze, design, implement, and maintain information systems.

Systems Development Life Cycle (SDLC)

- Systems Development Life Cycle (SDLC)

- Phases in SDLC:
  - Planning
  - Analysis
  - Design
  - Implementation
  - Maintenance

Standard and Evolutionary Views of SDLC

Figure 1-3 The systems development life cycle

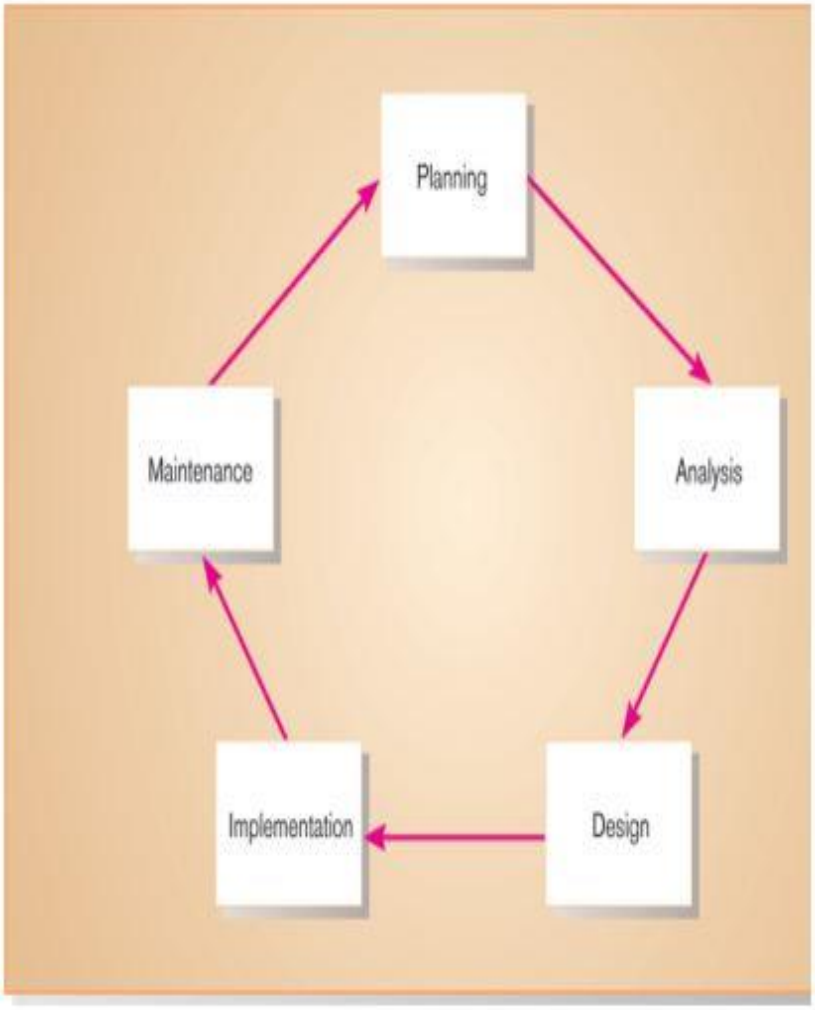
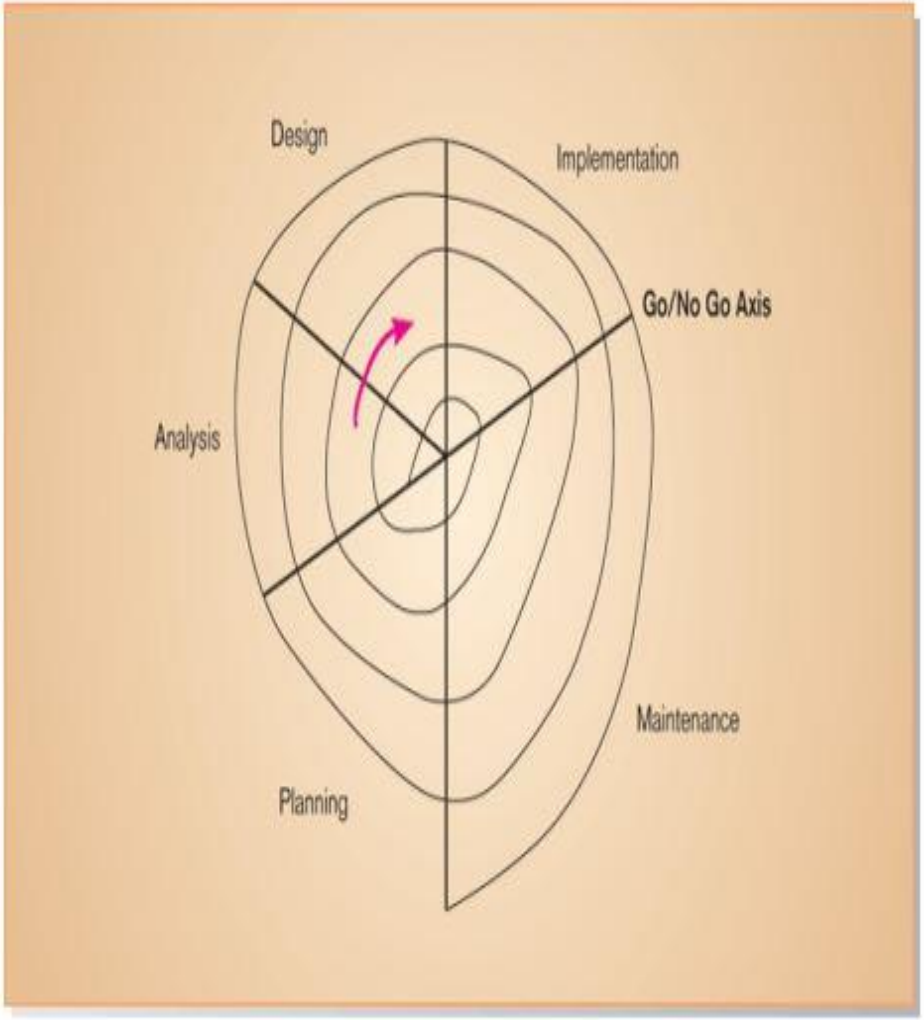
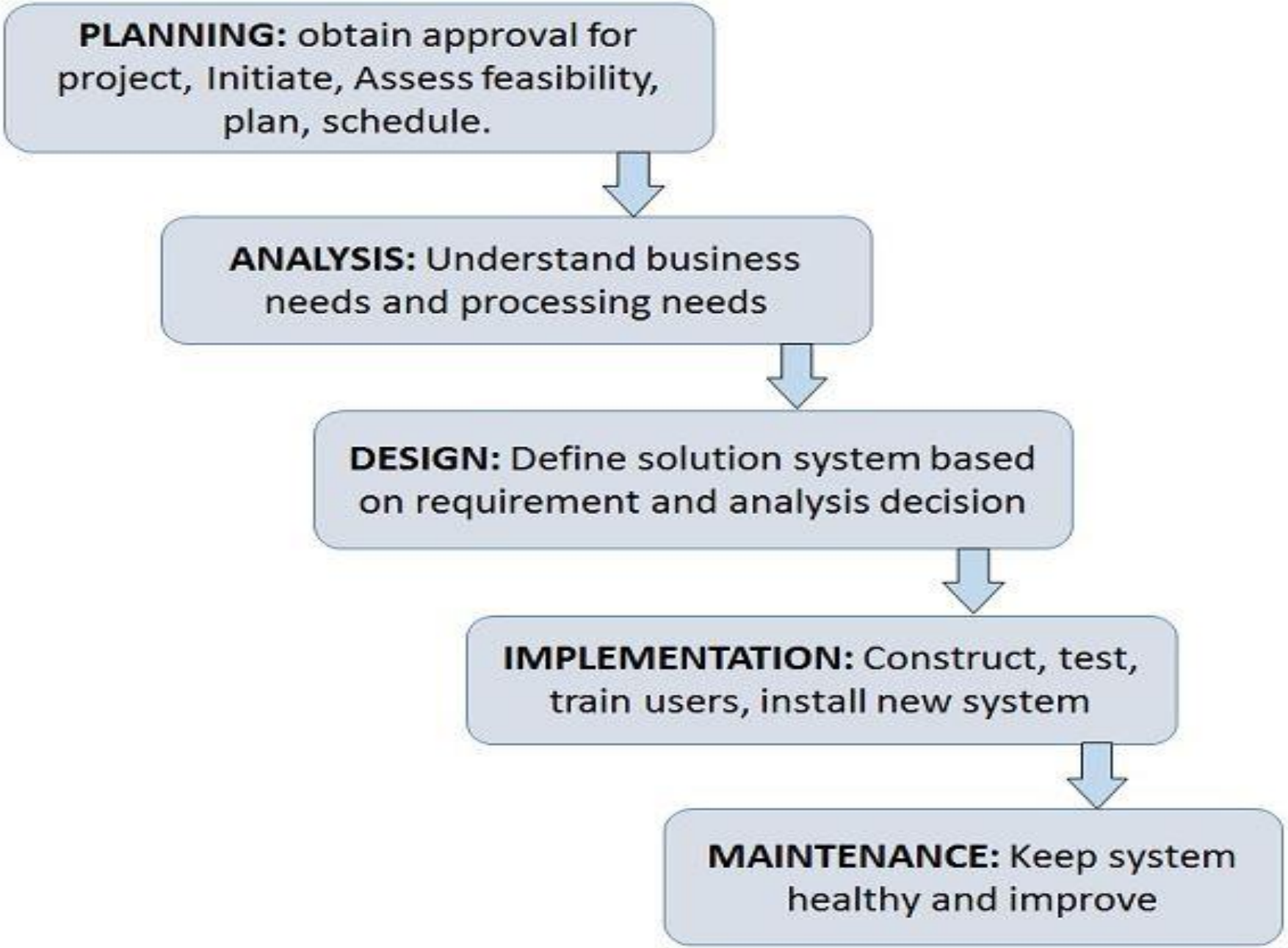


Figure 1-4 Evolutionary model SDLC




PLANNING: obtain approval for project, Initiate, Assess feasibility, plan, schedule.
ANALYSIS: Understand business needs and processing needs
DESIGN: Define solution system based on requirement and analysis decision
IMPLEMENTATION: Construct, test, train users, install new system
MAINTENANCE: Keep system healthy and improve

Systems Development Life Cycle (SDLC) (Cont.)

- Planning – an organization's total information system needs are identified, analyzed, prioritized, and arranged. The outcome of the project identification and selection process is a determination of which system development project should be undertaken by the organization, at least in terms of initial study.
- Major activities during planning are: i) Investigation of the system problem or opportunity. ii) presentation of reasons why the system should or should not be developed by the organization. Determining the scope of the proposed system

- Also produce a specific plan for the proposed project the team will follow. This specifies the time and resources needed for execution.
- Also identify whether the costs of developing system would give benefit.
- The second phase in the SDLC is analysis. During this phase, the analyst thoroughly studies the organization's current procedures and the information systems used to perform organizational tasks.
- Analysis has two sub phases. The first is requirements determination. In this sub phase, analysts work with users to determine what the users want from a proposed system. The requirements determination process usually involves a careful study of any current systems, manual and computerized, that might be replaced or enhanced as part of the project.

- In the second part of analysis, analysts study the requirements and structure them according to their interrelationships and eliminate any redundancies.
- The output of the analysis phase is a description of (but not a detailed design for) the alternative solution recommended by the analysis team. Once the recommendation is accepted by those with funding authority, the analysts can begin to make plans to acquire any hardware and system software necessary to build or operate the system as proposed.

- **The third phase in the SDLC is design**. During design, analysts convert the description of the recommended alternative solution into logical and then physical system specifications. The analysts must design all aspects of the system, from input and output screens to reports, databases, and computer processes. The analysts must then provide the physical specifics of the system they have designed, either as a model or as detailed documentation, to guide those who will build the new system.
- That part of the design process that is independent of any specific hardware or software platform is referred to as logical design. Theoretically, the system could be implemented on any hardware and systems software. The idea is to make sure that the system functions as intended.

- Once the overall high-level design of the system is worked out, the analysts begin turning logical specifications into physical ones. This process is referred to as physical design. As part of physical design, analysts design the various parts of the system to perform the physical operations necessary to facilitate data capture, processing, and information output. This can be done in many ways, from creating a working model of the system to be implemented to writing detailed specifications describing all the different parts of the system and how they should be built.
- In many cases, the working model becomes the basis for the actual system to be used. During physical design, the analyst team must determine many of the physical details necessary to build the final system, from the programming language the system will be written in, to the database system that will store the data, to the hardware platform on which the system will run.

- Often the choices of language, database, and platform are already decided by the organization or by the client, and at this point these information technologies must be taken into account in the physical design of the system. The final product of the design phase is the physical system specifications in a form ready to be turned over to programmers and other system builders for construction.
- The fourth phase in the SDLC is implementation. The physical system specifications, whether in the form of a detailed model or as detailed written specifications, are turned over to programmers as the first part of the implementation phase. During implementation, analysts turn system specifications into a working system that is tested and then put into use. Implementation includes coding, testing, and installation.

- During coding, programmers write the programs that make up the system. Sometimes the code is generated by the same system used to build the detailed model of the system.
- During testing, programmers and analysts test individual programs and the entire system in order to find and correct errors.
- During installation, the new system becomes part of the daily activities of the organization. Application software is installed, or loaded, on existing or new hardware, and users are introduced to the new system and trained. Testing and installation should be planned for as early as the project initiation and planning phase; both testing and installation require extensive analysis in order to develop exactly the right approach.

- Finalization of documentation, training programs
- Note that documentation and training programs are finalized during implementation; documentation is produced throughout the life cycle.
- The fifth and final phase in the SDLC is maintenance. When a system (including its training, documentation, and support) is operating in an organization, users sometimes find problems with how it works and often think of better ways to perform its functions. Also, the organization's needs with respect to the system change over time.
- In maintenance, programmers make the changes that users ask for and modify the system to reflect evolving business conditions. These changes are necessary to keep the system running and useful.

- In a sense, maintenance is not a separate phase but a repetition of the other life cycle phases required to study and implement the needed changes. The amount of time and effort devoted to maintenance depends a great deal on the performance of the previous phases of the life cycle.
- When an information system is no longer performing as desired, when maintenance costs become prohibitive, or when an organization's needs have changed substantially.
- Such problems indicate that it is time to begin designing the system's replacement, there by completing the loop and starting the life cycle over again.

Summary(SDLC)

Feasibility Study or Planning

- ✓ Define the problem and scope of existing system.
- ✓ Overview the new system and determine its objectives.
- ✓ Confirm project feasibility and produce the project Schedule.
- ✓ During this phase, threats, constraints, integration and security of
- ✓ system are also considered.
- ✓ A feasibility report for the entire project is created at the end of this
- ✓ phase.

Analysis and Specification

- ✓ Gather, analyze, and validate the information.
- ✓ Define the requirements and prototypes for new system.
- ✓ Evaluate the alternatives and prioritize the requirements.
- ✓ Examine the information needs of end-user and enhances the
- ✓ system goal.
- ✓ A Software Requirement Specification (SRS) document, which
- ✓ specifies the software, hardware, functional, and network
- ✓ requirements of the system is prepared at the end of this phase.

Summary(SDLC)

System Design

- ✓ Includes the design of application, network, databases, user interfaces, and system interfaces.
- ✓ Transform the SRS document into logical structure, which contains detailed and complete set of specifications that can be implemented in a programming language.
- ✓ Create a contingency, training, maintenance, and operation plan.
- ✓ Review the proposed design. Ensure that the final.

Implementation

- ✓ Implement the design into source code through coding.
- ✓ Combine all the modules together into training environment that detects errors and defects.
- ✓ A test report which contains errors is prepared through test plan that includes test related tasks such as test case generation, testing criteria, and resource allocation for testing.
- ✓ Integrate the information system into its environment and install the new system..

Summary(SDLC)

- **Maintenance/Support**

- ✓ Include all the activities such as phone support or physical on-site support for users that is required once the system is installing.
- ✓ Implement the changes that software might undergo over a period of time, or implement any new requirements after the software is deployed at the customer location.
- ✓ It also includes handling the residual errors and resolve any issues that may exist in the system even after the testing phase.
- ✓ Maintenance and support may be needed for a longer time for large systems and for a short time for smaller systems.

Traditional Waterfall SDLC

- Treat each phase as complete unto itself, never to be revisited once finished.
- Feedback came to be ignored in implementation.
- Traditionally, one phase ended and another began.
- System requirements “locked in” after being determined (can't change). Once the milestone had been reached and the new phase initiated, it became difficult to go back. The enormous amount of effort and time necessary to implement a specific design meant that it would be very expensive to make changes in a system once it was developed.
- Limited user involvement (only in requirements phase).

- Yet another criticism of the traditional waterfall SDLC is that the role of system users or customers was narrowly defined. User roles were often relegated to the requirements determination or analysis phases of the project, where it was assumed that all of the requirements could be specified in advance. Such an assumption, coupled with limited user involvement, reinforced the tendency of the waterfall model to lock in requirements too early, even after business conditions had changed.

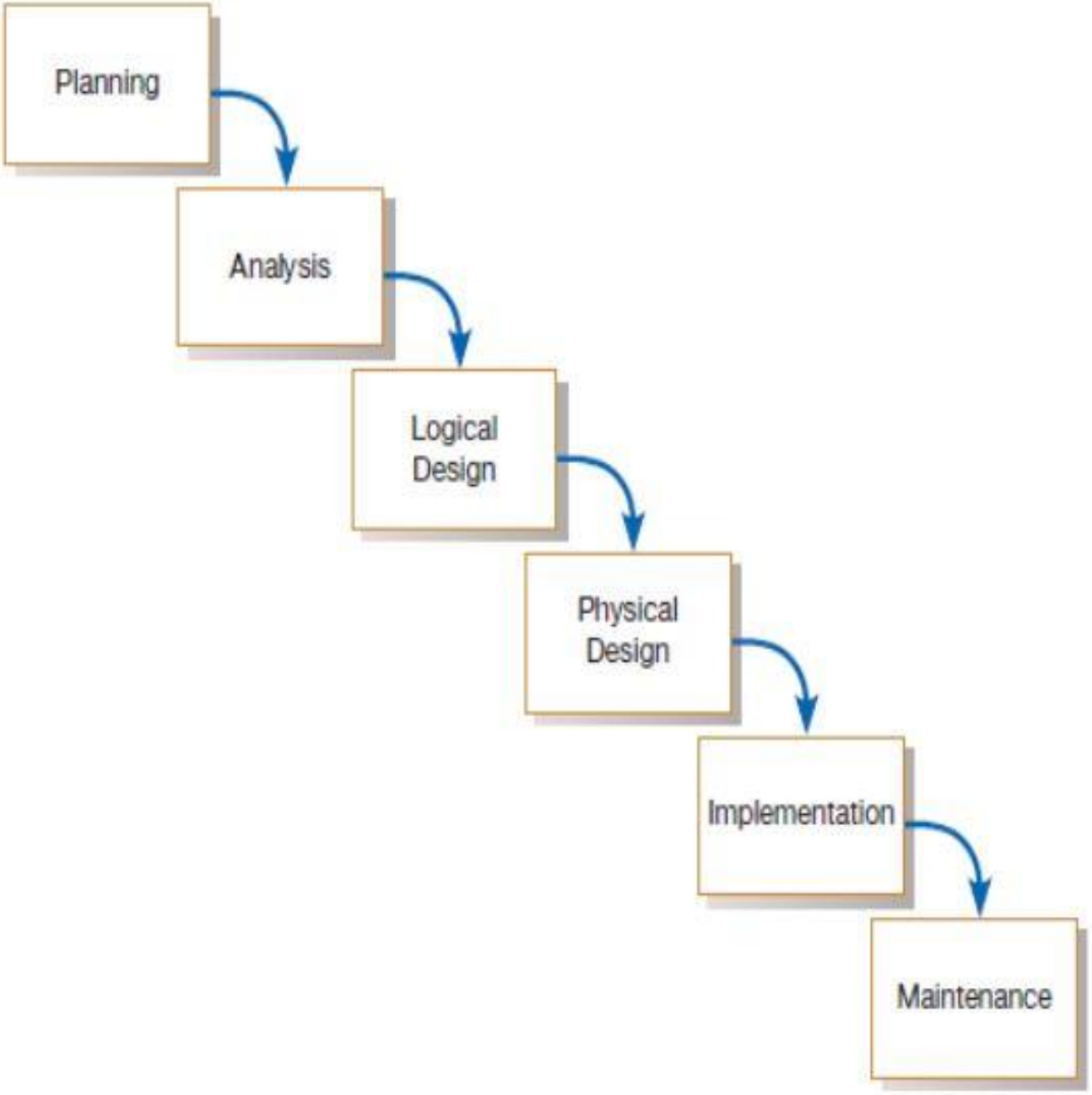


FIGURE 1-10
Traditional waterfall SDLC

Summary

- The Waterfall Model is the simple and classical model of all the model we have.
- This model is also known as linear sequential model.
- This model is theoretically model not a practical model.
- In this Model each and every phase must be completed before moving to the next phase,
- This model is suitable for the small project where the technical issues are very clear.

Waterfall Model - Application

- Every software developed is different and requires a suitable SDLC approach to be followed based on the internal and external factors. Some situations where the use of Waterfall model is most appropriate are:

I. Requirements are very well documented, clear and fixed.
II. Product definition is stable.
III. Technology is understood and is not dynamic.
IV. There are no ambiguous(Not Clear) requirements.
V. Ample(Enough) resources with required expertise are available to support the product.
VI. The project is short.

Advantage

- Some of the major advantages of the Waterfall Model are as follows
- Simple and easy to understand and use
- Easy to manage due to the rigidity of the model. Each phase has specific deliverables and a review process.
- Phases are processed and completed one at a time.
- Works well for smaller projects where requirements are very well understood.
- Clearly defined stages.
- Well understood milestones.
- Easy to arrange tasks.
- Process and results are well documented

Disadvantages

- No working software is produced until late during the life cycle.
- High amounts of risk and uncertainty.
- Not a good model for complex and object-oriented projects.
- Poor model for long and ongoing projects.
- Not suitable for the projects where requirements are at a moderate to high risk of changing. So, risk and uncertainty is high with this process model.
- It is difficult to measure progress within stages.
- Cannot accommodate changing requirements.
- Adjusting scope during the life cycle can end a project.
- Integration is done as a "big-bang. at the very end, which doesn't allow identifying any technological or business bottleneck or challenges early.

Different Approaches to Improving Development

- CASE TOOLS
- Rapid Application Development
- Service-Oriented Architecture
- Agile Methodologies
- extreme Programming
- Object-Oriented Analysis And Design

CASE TOOLS

- CASE stands for Computer Aided Software Engineering. It means, development and maintenance of software projects with help of various automated software tools.
- CASE tools are set of software application programs, which are used to automate SDLC activities. CASE tools are used by software project managers, analysts and engineers to develop software system.
- CASE tools can be used to help in multiple phases of the SDLC: Project Identification and Selection, analysis, design, and implementation and maintenance.
- There are number of CASE tools available to simplify various stages of Software Development Life Cycle such as Analysis tools, Design tools, Project management tools, Database Management tools, Documentation tools
- Use of CASE tools accelerates the development of project to produce desired result and helps to uncover flaws before moving ahead with next stage in software development.

The general types of CASE tools are listed below:

- Central Repository - CASE tools require a central repository, which can serve as a source of common, integrated and consistent information. Central repository is a central place of storage where product specifications, requirement documents, related reports and diagrams, other useful information regarding management is stored. Central repository also serves as data dictionary.
- Upper Case Tools - Upper CASE tools are used in planning, analysis and design stages of SDLC.
- Lower Case Tools - Lower CASE tools are used in implementation, testing and maintenance.
- Integrated Case Tools - Integrated CASE tools are helpful in all the stages of SDLC, from Requirement gathering to Testing and documentation. CASE tools can be grouped together if they have similar functionality, process activities and capability of getting integrated with other tools.

Examples

- Requirement Analysis Tool(flowchart maker, jira, basecamp etc.)
- Software Design Tool(animated software design, UML).
- Code Generation Tool(Cscope, eclipse,Github).
- Test Case Generation Tool(selenium, cucumber).
- Document Production Tool(Github, Zendesk, typora).
- Reverse Engineering Tool(IDA Pro, Ghidra, Cerbero Suite).

- Diagram tools : These tools are used to represent system components, data and control flow among various software components and system structure in a graphical form. For example, Flow Chart Maker tool for creating state-of-the-art flowcharts.

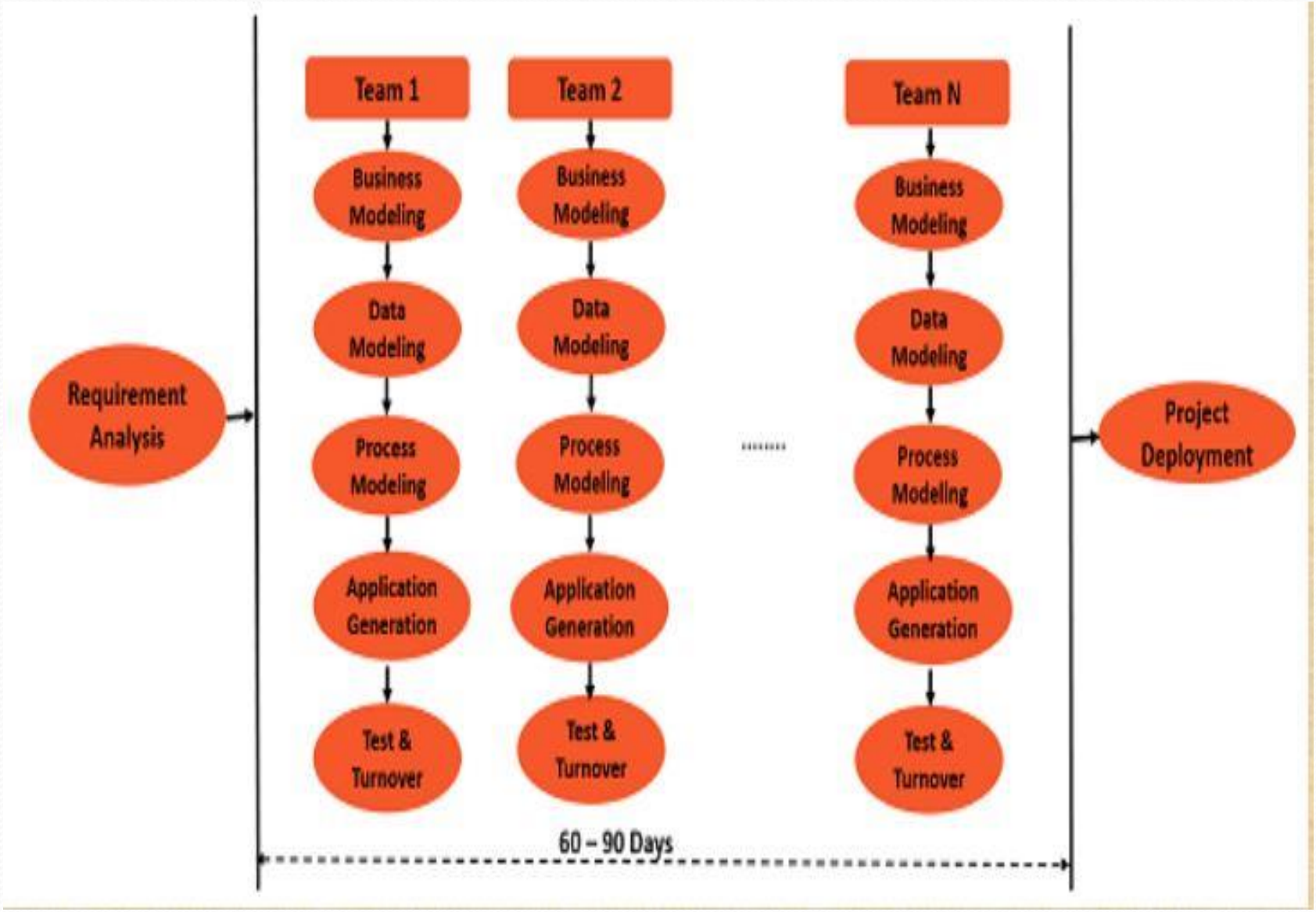
- Design Tools : These tools help software designers to design the block structure of the software, which may further be broken down in smaller modules using refinement techniques. These tools provides detailing of each module and interconnections among modules. For example, Animated Software Design

TABLE 1-2 Examples of CASE Usage within the SDLC

SDLC Phase	Key Activities	CASE Tool Usage
Project identification and selection	Display and structure high-level organizational information	Diagramming and matrix tools to create and structure information
Project initiation and planning	Develop project scope and feasibility	Repository and documentation generators to develop project plans
Analysis	Determine and structure system requirements	Diagramming to create process, logic, and data models
Logical and physical design	Create new system designs	Form and report generators to prototype designs; analysis and documentation generators to define specifications
Implementation	Translate designs into an information system	Code generators and analysis, form and report generators to develop system; documentation generators to develop system and user documentation
Maintenance	Evolve information system	All tools are used (repeat life cycle)

RAD(Rapid Application Development)

- Rapid application development is a software development methodology that uses minimal planning in favor of rapid prototyping. A prototype is a working model that is functionally equivalent to a component of the product.
- In the RAD model, the functional modules are developed in parallel as prototypes and are integrated to make the complete product for faster product delivery. Since there is no detailed preplanning, it makes it easier to incorporate the changes within the development process.
- The most important aspect for this model to be successful is to make sure that the prototypes developed are reusable.


Requirement Analysis
Team 1
Business Modeling
Data Modeling
Process Modeling
Application Generation
Test & Turnover
Team 2
Business Modeling
Data Modeling
Process Modeling
Application Generation
Test & Turnover
.....
Team N
Business Modeling
Data Modeling
Process Modeling
Application Generation
Test & Turnover
Project Deployment
60 – 90 Days

- ## Advantages of the RAD model:

- Reduced development time.
- Increases reusability of components
- Quick initial reviews occur
- Encourages customer feedback
- Integration from very beginning solves a lot of integration issues.

- ## Disadvantages of RAD model:

- Depends on strong team and individual performances for identifying business requirements.
- Only system that can be modularized can be built using RAD
- Requires highly skilled developers/designers.
- High dependency on modeling skills
- Inapplicable to cheaper projects as cost of modeling and automated code generation is very high.

- **When to use RAD model:**

✓ RAD should be used when there is a need to create a system that can be modularized in 2-3 months of time.

✓ It should be used if there's high availability of designers for modeling and the budget is high enough to afford their cost along with the cost of automated code generating tools.

✓ RAD SDLC model should be chosen only if resources with high business knowledge are available and there is a need to produce the system in a short span of time (2-3 months).

Service-Oriented Architecture:

- It is modern and new concept of software development . It make individual SOA services are unassociated or loosely coupled to another. Each service executes one action. Each service can be used in other application within the organization or even in other organizations.

- We can say that service-oriented architecture is simply a group of services that can be called upon to provide specific functions. Rather than including calls to other services, a service can use certain defined protocols so that it can communicate with other services.

Advantages of Service-oriented Architecture

- SOA allows reuse the service of an existing system alternately building the new system.
- It allows plugging in new services or upgrading existing services to place the new business requirements.
- It can enhance the performance, functionality of a service and easily makes the system upgrade.
- SOA has capability to adjust or modify the different external environments and large applications can be managed easily.
- The companies can develop applications without replacing the existing applications.
- It provides reliable applications in which you can test and debug the independent services easily as compared to large number of code.

Disadvantages of Service-oriented Architecture

- SOA requires high investment cost (means large investment on technology, development and human resource).
- There is greater overhead when a service interacts with another service which increases the response time and machine load while validating the input parameters.
- SOA is not suitable for GUI (graphical user interface) applications which will become more complex when the SOA requires the heavy data exchange.

SOA vs API

Agile Methodologies

- The word ‘agile’ means – Able to move your body quickly and easily.
- In software development, the term ‘agile’ is adapted to mean ‘the ability to respond to changes – changes from Requirements, Technology and People.
- According to Fowler (2003), the Agile Methodologies share three key principles: (1) a focus on adaptive rather than predictive methodologies, (2) a focus on people rather than roles, and (3) a focus on self-adaptive processes.
- Agile Methodologies share iterative development (Martin, 1999).Iterative development focuses on the frequent production of working versions of a system that have a subset of the total number of required features. Iterative development provides feedback to customers and developers alike.

- Agile Methodologies are not for every project. Fowler (2003) recommends an agile or adaptive process if your project involves
  - Unpredictable or dynamic requirements,
  - Responsible and motivated developers,
  - Customers who understand the process and will get involved.

Extreme Programming

- Extreme Programming is a lightweight, efficient, low-risk, flexible, predictable, scientific, and fun way to develop a software.
- eXtreme Programming (XP) was conceived and developed to address the specific needs of software development by small teams in the face of vague(indefinite) and changing requirements.
- Extreme Programming is one of the Agile software development methodologies. It provides values and principles to guide the team behavior.
- Why is it called “Extreme?”
  - Extreme Programming takes the effective principles and practices to extreme levels.
  - Code reviews are effective as the code is reviewed all the time.
  - Testing is effective as there is continuous regression and testing.
  - Design is effective as everybody needs to do refactoring daily.
  - Integration testing is important as integrate and test several times a day.

Object-Oriented Analysis And Design

- The object-oriented approach combines data and processes (called methods) into single entities called objects. Objects usually correspond to the real things an information system deals with, such as customers, suppliers, contracts, and rental agreements.
- The goal of OOAD is to make systems elements more reusable, thus improving system quality and the productivity of systems analysis and design.
- Another key idea behind object orientation is inheritance. Inheritance allows the creation of new classes that share some of the characteristics of existing classes.
- For example, from a class of objects called “person,” you can use inheritance to define another class of objects called “customer. "Objects of the class “customer” would share certain characteristics with objects of the class “person”: They would both have names, addresses, phone numbers, and so on. Because “person” is the more general class and “customer” is more specific, every customer is a person but not every person is a customer.

Object-Oriented Analysis And Design

- The object-oriented approach combines data and processes (called methods) into single entities called objects. Objects usually correspond to the real things an information system deals with, such as customers, suppliers, contracts, and rental agreements.
- The goal of OOAD is to make systems elements more reusable, thus improving system quality and the productivity of systems analysis and design.
- Another key idea behind object orientation is inheritance. Inheritance allows the creation of new classes that share some of the characteristics of existing classes.
- For example, from a class of objects called “person,” you can use inheritance to define another class of objects called “customer. "Objects of the class “customer” would share certain characteristics with objects of the class “person”: They would both have names, addresses, phone numbers, and so on. Because “person” is the more general class and “customer” is more specific, every customer is a person but not every person is a customer.

- The object-oriented approach to systems development shares the iterative development approach of the Agile Methodologies.
- One of the most popular realizations of the iterative approach for object-oriented development is the Rational Unified Process (RUP), which is based on an iterative, incremental approach to systems development. RUP has four phases: inception, elaboration, construction, and transition (see Figure).
- In the inception phase, analysts define the scope, determine the feasibility of the project, understand user requirements, and prepare a software development plan.
- In the elaboration phase, analysts detail user requirements and develop a baseline architecture. Analysis and design activities constitute the bulk of the elaboration phase. In the construction phase, the software is actually coded, tested, and documented.
- In the transition phase, the system is deployed, and the users are trained and supported.

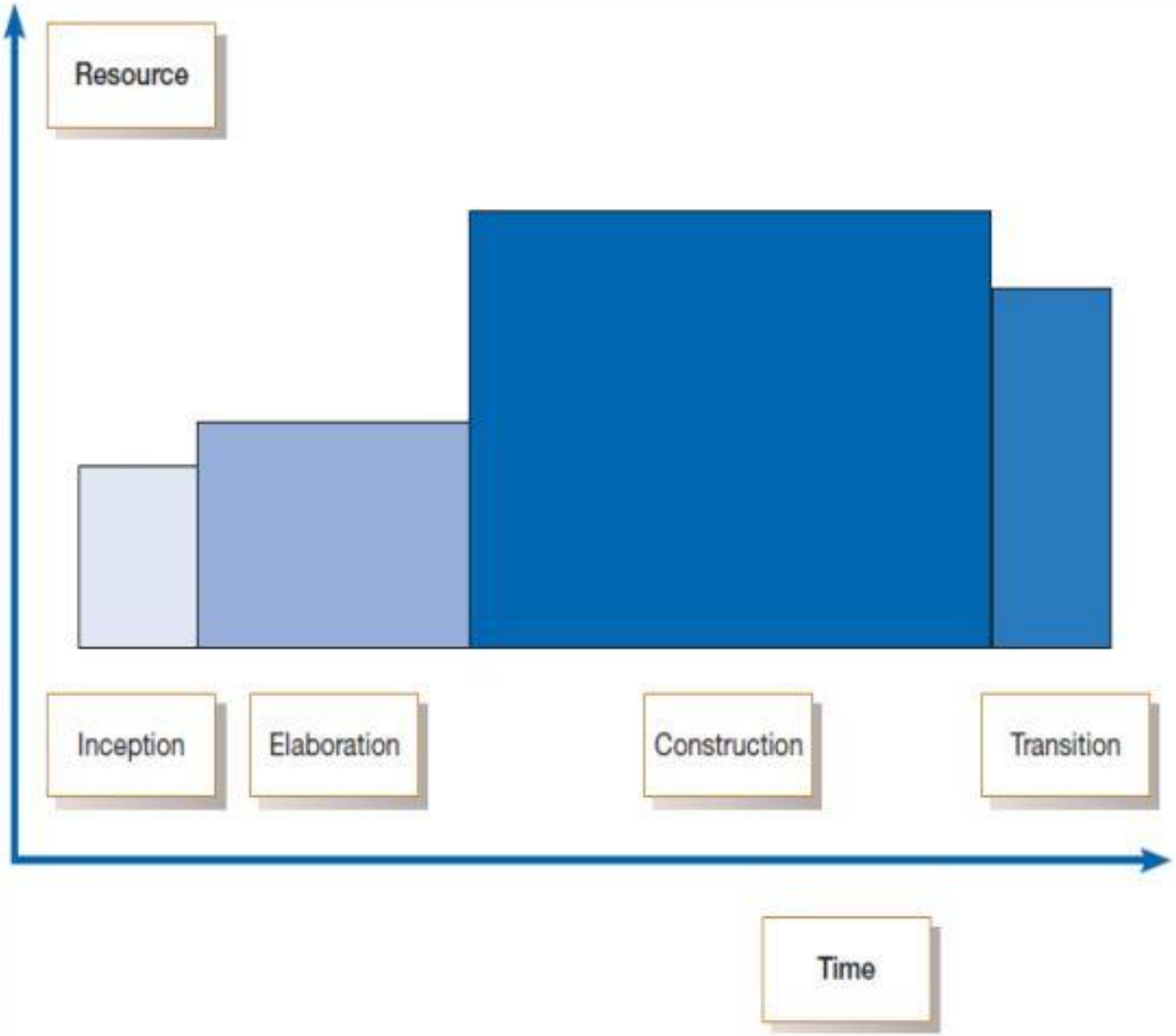


FIGURE 1-11
Phases of OOAD-based development

- The construction phase is generally the longest and the most resource intensive. The elaboration phase is also long, but less resource intensive. The transition phase is resource intensive but short. The inception phase is short and the least resource intensive. The areas of the rectangles in Figure 1-11 provide an estimate of the overall resources allocated to each phase.

The Origins Of Software

- Introduction: If you wanted to write application software, you did it in-house, and you wrote the software from scratch. Today there are many different sources of software and firms that produce software, rather than in the information systems department of a corporation. But for those of you who do go on to work in a corporate information systems department, the focus is no longer exclusively on in-house development.
- SYSTEMS ACQUISITION: Internal corporate information systems departments now spend a smaller and smaller proportion of their time and effort on developing systems from scratch. Companies continue to spend relatively little time and money on traditional software development and maintenance. Instead, they invest in packaged software, open-source software, and outsourced services.

- Outsourcing: If one organization develops or runs a computer application for another organization, that practice is called outsourcing. Outsourcing includes a spectrum of working arrangements. At one extreme is having a firm develop and run your application on its computers—all you do is supply input and take output. A common example of such an arrangement is a company that runs payroll applications for clients so that clients do not have to develop an independent in-house payroll system. Instead, they simply provide employee payroll information to the company, and, for a fee, the company returns completed paychecks, payroll accounting reports, and tax and other statements for employees. For many organizations, payroll is a very cost-effective operation when outsourced in this way.
- Outsourcing is big business. Some organizations outsource the information technology (IT) development of many of their IT functions at a cost of billions of dollars. Most organizations outsource at least some aspect of their information systems activities.

- Some reasons for outsourcing include
  - freeing up internal resources,
  - increasing the revenue potential of the organization,
  - reducing time to market,
  - increasing process efficiencies, and
  - outsourcing noncore activities

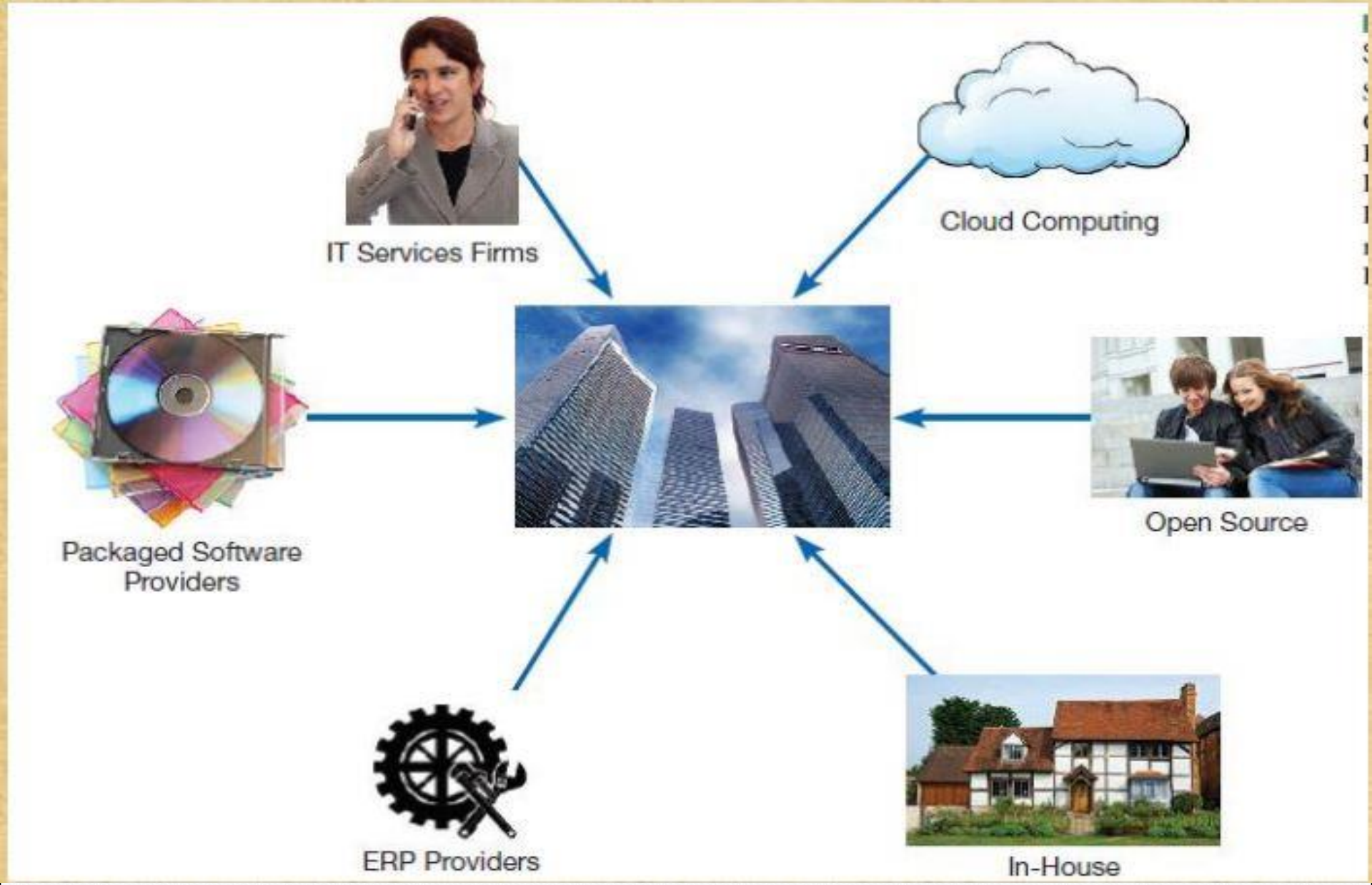


Figure: Source of Application Software

Sources of Software:

- We can group the sources of software into six major categories: information technology services firms, packaged software producers, enterprise-wide solutions, cloud computing vendors, open-source software, and in-house developers (As shown in Figure).
- Information Technology Services Firms: If a company needs an information system but does not have the expertise or the personnel to develop the system in-house the company will likely consult an information technology services firm. IT services firms help companies develop custom information systems for internal use, or they develop, host, and run applications for customers, or they provide other services. Note in Table 2-1 that many of the leading software companies in the world specialize in services, which include custom systems development. These firms employ people with expertise in the development of information systems. Their consultants may also have expertise in a given business area. For example, consultants who work with banks understand financial institutions as well as information systems. Consultants use many of the same methodologies, techniques, and tools that companies use to develop systems in-house.

TABLE 2-1 Leading Software Firms and Their Development Specializations

Specialization	Example Firms or Websites
IT Services	Accenture Computer Sciences Corporation (CSC) IBM HP
Packaged Software Providers	Intuit Microsoft Oracle SAP AG Symantec
Enterprise Software Solutions	Oracle SAP AG
Cloud Computing	Amazon.com Google IBM Microsoft Salesforce.com
Open Source	SourceForge.net

- Packaged Software Producers: The growth of the software industry has been phenomenal since its beginnings in the mid 1960s. Some of the largest computer companies in the world are companies that produce software exclusively. A good example is Microsoft, probably the best-known software company in the world.
- Almost 87 percent of Microsoft's revenue comes from its software sales, mostly for its Windows operating systems and its personal productivity software, the Microsoft Office Suite. Also listed in Table 2-1, Oracle is exclusively a software company known primarily for its database software, but Oracle also makes enterprise systems.
- Software companies develop software to run on many different computer platforms, from microcomputers to large mainframes.
- •Software companies consult with system users after the initial software design has been completed and an early version of the system has been built. The systems are then tested in actual organizations to determine whether there are any problems or if any improvements can be made. Until testing is completed, the system is not offered for sale to the public.

Enterprise Solutions Software

- Many firms have chosen complete software solutions, called enterprise solutions or enterprise resource planning (ERP) systems, to support their operations and business processes. These ERP software solutions consist of a series of integrated modules. Each module supports an individual, traditional business function, such as accounting, distribution, manufacturing, or human resources. The difference between the modules and traditional approaches is that the modules are integrated to focus on business processes rather than on business functional areas. For example, a series of modules will support the entire order entry process, from receiving an order, to adjusting inventory, to shipping to billing, to after-the-sale service. The traditional approach would use different systems in different functional areas of the business, such as a billing system in accounting and an inventory system in the warehouse. Using enterprise software solutions, a firm can integrate all parts of a business process in a unified information system. All aspects of a single transaction occur seamlessly within a single information system, rather than as a series of disjointed, separate systems focused on business functional areas.

- The benefits of the enterprise solutions approach include a single repository of data for all aspects of a business process and the flexibility of the modules. A single repository ensures more consistent and accurate data, as well as less maintenance.
- The modules are flexible because additional modules can be added as needed once the basic system is in place. Added modules are immediately integrated into the existing system. However, there are disadvantages to enterprise solutions software.
- The systems are very complex, so implementation can take a long time to complete. Organizations typically do not have the necessary expertise in-house to implement the systems, so they must rely on consultants or employees of the software vendor, which can be very expensive. In some cases, organizations must change how they do business in order to benefit from a migration to enterprise solutions.

- Cloud Computing : Another method for organizations to obtain applications is to rent them or license them from third-party providers who run the applications at remote sites. Users have access to the applications through the Internet or through virtual private networks. The application provider buys, installs, maintains, and upgrades the applications. Users pay on a per-use basis or they license the software, typically month to month. Although this practice has been known by many different names over the years, today it is called cloud computing. Cloud computing refers to the provision of applications over the Internet, where customers do not have to invest in the hardware and software resources needed to run and maintain the applications.
- You may have seen the Internet referred to as a cloud in other contexts, which comes from how the Internet is depicted on computer network diagrams. A well-known example of cloud computing is Google Apps, where users can share and create documents, spreadsheets, and presentations (Figure 2-4). Another well-known example is Salesforce.com, which provides customer relationship management software online. Cloud computing encompasses many areas of technology, including software as a service (often referred to as SaaS), which includes Salesforce.com, and hardware as a service, which includes Amazon Web Services and allows companies to order server capacity and storage on demand.

- Taking the cloud computing route has its advantages. The top three reasons for choosing to go with cloud computing, all of which result in benefits for the company, are

- Freeing internal IT staff,
- Gaining access to applications faster than via internal development,
- Achieving lower cost access to corporate-quality applications.

- Getting your computing through a cloud also makes it easier to walk away from an unsatisfactory systems solution. Other reasons include cost effectiveness, speed to market, and better performance .
- IT managers do have some concerns about cloud computing, however. The primary concern is over security. Concerns over security are based on storing company data on machines one does not own and that others can access. In fact, the top two reasons for not using cloud services are concerns about unauthorized access to proprietary information and unauthorized access to customer information's. Another concern is reliability. Some warn that the cloud is actually a network of networks, it is vulnerable to unexpected risks due to its complexity .

Open-Source Software

- Open-source software is unlike the other types of software you have read about so far. Open-source software is different because it is freely available, not just the final product but the source code itself. It is also different because it is developed by a community of interested people instead of by employees of a particular company. Open-source software performs the same functions as commercial software, such as operating systems, e-mail, database systems, web browsers, and so on. Some of the most well-known and popular open-source software names are Linux, an operating system; mySQL, a database system; and Firefox, a web browser. Open source also applies to software components and objects. Open source is developed and maintained by communities of people, and sometimes these communities can be very large.
- If the software is free, you might wonder how anybody makes any money by developing open-source software. Companies and individuals can make money with open source in two primary ways:

1. by providing maintenance and other services or
2. by providing one version of the software free and selling a more fully featured version.

- Some open-source solutions have more of an impact on the software industry than others. Linux, for example, has been very successful in the server software market, where it is estimated to have as much as 36 percent of the market share (W3Techs, 2015). In the desktop operating systems market, Linux has about 1 percent market share. Other open-source software products, such as mySQL, have also been successful, and open source's share of the software industry seems destined to continue to grow.

In-House Development:

- We have talked about several different types of external organizations that serve as sources of software, but in-house development remains an option. In-house development has become a progressively smaller piece of all systems development work that takes place in and for organizations. As you read earlier in this chapter, internal corporate information systems departments now spend a smaller and smaller proportion of their time and effort on developing systems from scratch. In-house development can lead to a larger maintenance burden than other development methods, such as packaged applications. A study by Banker, Davis, and Slaughter found that using a code generator as the basis for in-house development was related to an increase in maintenance hours, whereas using packaged applications was associated with a decrease in maintenance effort.

- Of course, in-house development need not entail (to make something necessary) development of all of the software that will constitute the total system. Hybrid solutions involving some purchased and some in-house software components are common. If you choose to acquire software from outside sources, this choice is made at the end of the analysis phase.
- The choice between a package and an external supplier will be determined by your needs, not by what the supplier has to sell. As we will discuss, the results of your analysis study will define the type of product you want to buy and will make working with an external supplier much easier, more productive, and worthwhile. Table 2-2 compares the six different software sources discussed in this section.

TABLE 2-2 Comparison of Six Different Sources of Software Components

Producers	When to Go to This Type of Organization for Software	Internal Staffing Requirements
IT services firms	When task requires custom support and system can't be built internally or system needs to be sourced	Internal staff may be needed, depending on application
Packaged software producers	When supported task is generic	Some IS and user staff to define requirements and evaluate packages
Enterprise-wide solutions vendors	For complete systems that cross functional boundaries	Some internal staff necessary but mostly need consultants
Cloud computing	For instant access to an application; when supported task is generic	Few; frees up staff for other IT work
Open-source software	When supported task is generic but cost is an issue	Some IS and user staff to define requirements and evaluate packages
In-house developers	When resources and staff are available and system must be built from scratch	Internal staff necessary though staff size may vary

REUSE

- Reuse is the use of previously written software resources in new applications. Because so many bits and pieces of applications are relatively generic across applications, it seems intuitive that great savings can be achieved in many areas if those generic bits and pieces do not have to be written a new each time they are needed. Reuse should increase programmer productivity because being able to use existing software for some functions means they can perform more work in the same amount of time. Reuse should also decrease development time, minimizing schedule overruns. Because existing pieces of software have already been tested, reusing them should also result in higher-quality software with lower defect rates, decreasing maintenance costs.
- Advantages: Less effort, Time-saving, Reduce cost, Increase software productivity, Utilize fewer resources, Leeds to a better quality software.

Managing the information system project

- • Introduction
- In this chapter, we focus on the systems analyst's role as project manager of an information systems project. Throughout the SDLC, the project manager is responsible for initiating, planning, executing, and closing down the systems development project. Project management is arguably the most important aspect of an information systems development project. Effective project management helps to ensure that systems development projects meet customer expectations and are delivered within budget and time constraints.

- Today, there is a shift in the types of projects most firms are undertaking, which makes project management much more difficult and even more critical to project success. For example, in the past, organizations focused much of their development on very large, custom designed, stand-alone applications. Today, much of the systems development effort in organizations focuses on implementing packaged software such as enterprise resource planning (ERP) and data warehousing systems. Existing legacy applications are also being modified so that business-to-business transactions can occur seamlessly over the Internet. New web-based interfaces are being added to existing legacy systems so that a broader range of users, often distributed globally, can access corporate information and systems. Additionally, software developed by global outsourcing partners that must be integrated into an organization's existing portfolio of applications is now common practice (Overby, 2013). Working with vendors to supply applications, with customers or suppliers to integrate systems, or with a broader and more diverse user community requires that project managers be highly skilled. Consequently, it is important that you gain an understanding of the project management process; this will become a critical skill for your future success.

- The project management process involves four phases:
  - Initiating the project
  - Planning the project
  - Executing the project
  - Closing down the project

Initiating a project:

During project initiation, the project manager performs several activities to assess the size, scope, and complexity of the project and to establish procedures to support subsequent activities. The types of activities that will perform when initiating a project are summarized below:

1. Establishing the project initiation team: This activity involves organizing project team members to assist in accomplishing the project initiation activities. (For example, during the Purchasing Fulfillment System project at PVF, Chris Martin was assigned to support the Purchasing department. It is a PVF policy that all initiation teams consist of at least one user representative, in this case Juanita Lopez, and one member of the information systems (IS) development group. Therefore, the project initiation team consisted of Chris and Juanita; Chris was the project manager.)

2. **Stablishing a relationship with the customer** : A thorough understanding of your customer builds stronger partnerships and higher levels of trust. (At PVF, management has tried to foster strong working relationships between business units (like Purchasing) and the IS development group by assigning a specific individual to work as a liaison between both groups. Because Chris had been assigned to the Purchasing unit for some time, he was already aware of some of the problems with the existing purchasing systems. PVF's policy of assigning specific individuals to each business unit helped to ensure that both Chris and Juanita were comfortable working together prior to establishing relationships with customers.)

3. **Establishing the project initiation plan**: This step defines the activities required to organize the initiation team while it is working to define the goals and scope of the project.

4. **Establishing management procedures** : Successful projects require the development of effective management procedures.

5. **Establishing the project management environment and project workbook** :The focus of this activity is to collect and organize the tools that you will use while managing the project and to construct the project workbook. Diagrams, charts, and system descriptions provide much of the project workbook contents. Thus, the project workbook serves as a repository for all project correspondence, inputs, outputs, deliverables, procedures, and standards established by the project team.

6. **Developing the project charter:** The project charter is a short (typically one page), high-level document prepared for the customer that describes what the project will deliver and outlines many of the key elements of the project

Planning the project

- Research has found a positive relationship between effective project planning and positive project outcomes. Project planning involves defining clear, discrete activities and the work needed to complete each activity within a single project. It often requires you to make numerous assumptions about the availability of resources such as hardware, software, and personnel. It is much easier to plan nearer-term activities than those occurring in the future. (In actual fact, you often have to construct longer-term plans that are more general in scope and nearer term plans that are more detailed. The repetitive nature of the project management process requires that plans be constantly monitored throughout the project and periodically updated (usually after each phase), based upon the most recent information.)

Project Planning

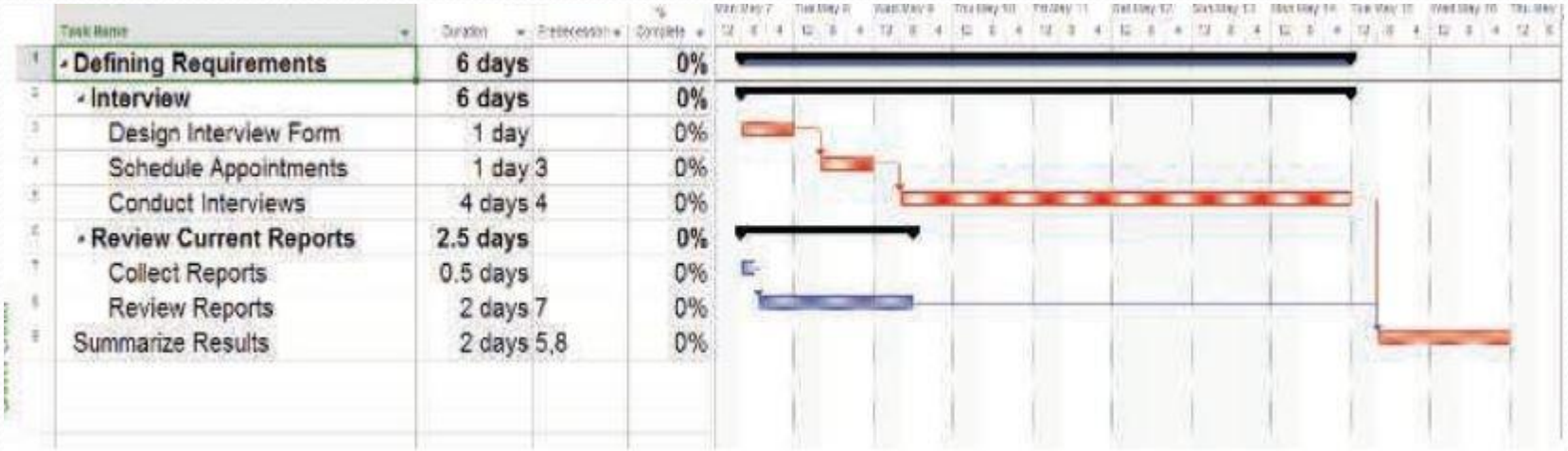
1. Describing Project Scope, Alternatives, and Feasibility
2. Dividing the Project into Manageable Tasks
3. Estimating Resources and Creating a Resource Plan
4. Developing a Preliminary Schedule
5. Developing a Communication Plan
6. Determining Project Standards and Procedures
7. Identifying and Assessing Risk
8. Creating a Preliminary Budget
9. Developing a Project Scope Statement
10. Setting a Baseline Project Plan

- 1. **Describing project scope, alternatives, and feasibility** • The purpose of this activity is to understand the content and complexity of the project. Within PVF's systems development methodology, one of the first meetings must focus on defining a project's scope. Although project scope information was not included in the SSR(System Service Request) developed by Chris and Juanita, it was important that both shared the same vision for the project before moving too far along. During this activity, you should reach agreement on the following questions:

  - What problem or opportunity does the project address?
  - What are the quantifiable results to be achieved?
  - What needs to be done?
  - How will success be measured?
  - How will we know when we are finished?

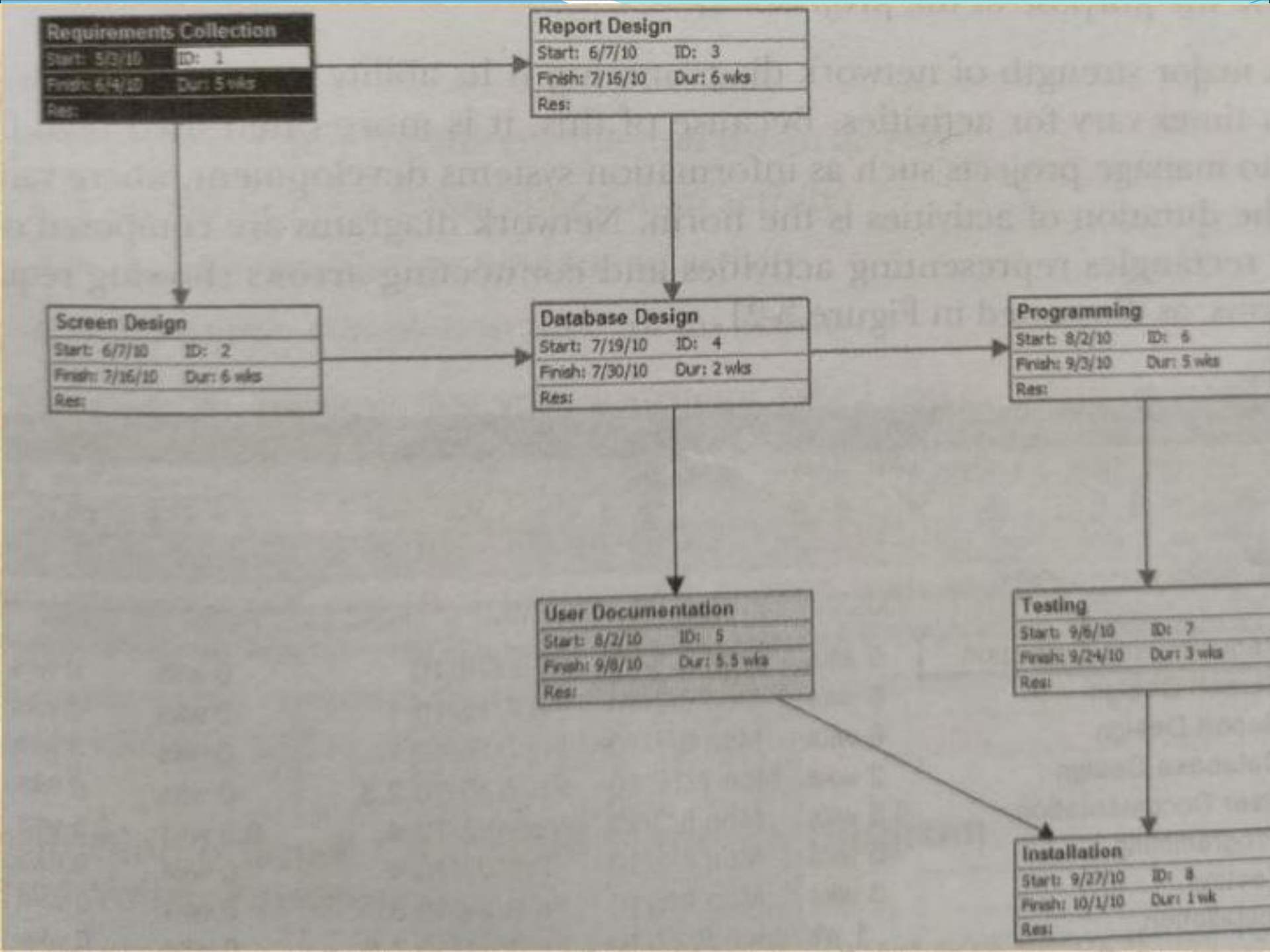
- After defining the scope of the project, your next objective is to identify and document general alternative solutions for the current business problem or opportunity. You must then assess (to judge or decide value or importance) the feasibility of each alternative solution and choose which to consider during subsequent SDLC phases.

- 2. Dividing the project into manageable tasks
- This is a critical activity during the project planning process. Here, you must divide the entire project into manageable tasks and then logically order them to ensure a smooth evolution between tasks.
- For example, suppose that you are working on a new development project and need to collect system requirements by interviewing users of the new system and reviewing reports they currently use to do their job. A work breakdown for these activities is represented in a Gantt chart in Figure . A Gantt chart is a graphical representation of a project that shows each task as a horizontal bar whose length is proportional to its time for completion. Different colors, shades, or shapes can be used to highlight each kind of task.



	Task Name	Duration	Predecessors	% Complete
1	Defining Requirements	6 days		0%
2	Interview	6 days		0%
3	Design Interview Form	1 day		0%
4	Schedule Appointments	1 day	3	0%
5	Conduct Interviews	4 days	4	0%
6	Review Current Reports	2.5 days		0%
7	Collect Reports	0.5 days		0%
8	Review Reports	2 days	7	0%
9	Summarize Results	2 days	5,8	0%

- **3. Estimating resources and creating a resource plan**
- The goal of this activity is to estimate resource requirements for each project activity and to use this information to create a project resource plan. The resource plan helps assemble and deploy resources in the most effective manner. For example, you would not want to bring additional programmers onto the project at a rate faster than you could prepare work for them. Project managers use a variety of tools to assist in making estimates of project size and costs. The most widely used method is called COCOMO (constructive cost model), COCOMO predict human resource requirements for basic, intermediate, and very complex systems.
- 4. **Developing a preliminary schedule**
- During this activity, you use the information on tasks and resource availability to assign time estimates to each activity in the work breakdown structure. These time estimates will enable you to create target starting and ending dates for the project. Target dates can be revisited and modified until a schedule is produced that is acceptable to the customer. Determining an acceptable schedule may require that you find additional or different resources or that the scope of the project be changed. The schedule may be represented as a Gantt chart or as a network diagram.


Requirements Collection
Start: 5/3/10 ID: 1
Finish: 6/4/10 Dur: 5 wks
Res:
Report Design
Start: 6/7/10 ID: 3
Finish: 7/16/10 Dur: 6 wks
Res:
Screen Design
Start: 6/7/10 ID: 2
Finish: 7/16/10 Dur: 6 wks
Res:
Database Design
Start: 7/19/10 ID: 4
Finish: 7/30/10 Dur: 2 wks
Res:
Programming
Start: 8/2/10 ID: 6
Finish: 9/3/10 Dur: 5 wks
Res:
User Documentation
Start: 8/2/10 ID: 5
Finish: 9/8/10 Dur: 5.5 wks
Res:
Testing
Start: 9/6/10 ID: 7
Finish: 9/24/10 Dur: 3 wks
Res:
Installation
Start: 9/27/10 ID: 8
Finish: 10/1/10 Dur: 1 wk
Res:

- 5. Developing a communication plan
- The goal of this activity is to outline the communication procedures among management, project team members, and the customer. The communication plan includes when and how written and oral reports will be provided by the team, how team members will coordinate work, what messages will be sent to announce the project to interested parties, and what kinds of information will be shared with vendors and external contractors involved with the project.
- When developing a communication plan, numerous questions must be answered in order to assure that the plan is comprehensive and complete, including the following:

- Who are the stakeholders for this project?
- What information does each stakeholder need?
- When, and at what interval, does this information need to be produced?
- What sources will be used to gather and generate this information?
- Who will collect, store, and verify the accuracy of this information?
- Who will organize and package this information into a document?
- Who will be the contact person for each stakeholder should any questions arise?
- What format will be used to package this information?

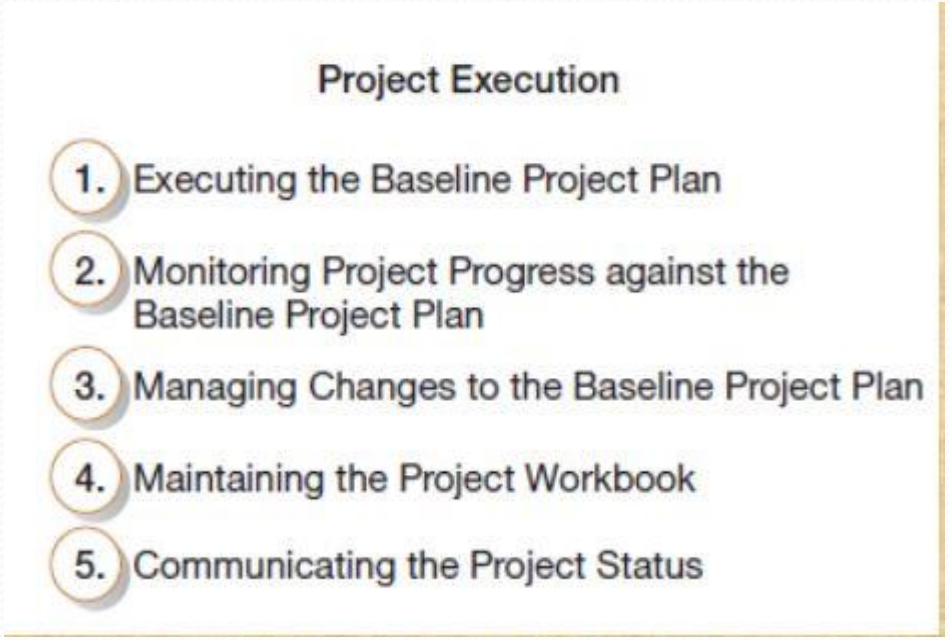
- What communication medium will be most effective for delivering this information to the stakeholder?
- Once these questions are answered for each stakeholder, a comprehensive communication plan can be developed. In this plan, a summary of communication documents, work assignments, schedules, and distribution methods will be outlined.
- **6. Determining project standards and procedures**
- During this activity, you will specify how various deliverables are produced and tested by you and your project team. For example, the team must decide which tools to use, how the standard SDLC might be modified, which SDLC methods will be used, documentation styles (e.g., type fonts and margins for user manuals), how team members will report the status of their assigned activities, and terminology. Setting project standards and procedures for work acceptance is a way to ensure the development of a high-quality system. Also, it is much easier to train new team members when clear standards are in place. Organizational standards for project management and conduct make the determination of individual project standards easier and the interchange or sharing of personnel among different projects feasible.

- **7. Identifying and assessing risk**
- The goal of this activity is to identify sources of project risk and estimate the consequences of those risks. Risks might arise from the use of new technology, prospective users' resistance to change, availability of critical resources, competitive reactions or changes in regulatory actions due to the construction of a system, or team member inexperience with technology or the business area. You should continually try to identify and assess project risk.
- 8. Creating a preliminary budget
- During this phase, you need to create a preliminary budget that outlines the planned expenses and revenues associated with your project. The project justification will demonstrate that the benefits are worth these costs.
- **9. Developing a Project Scope Statement**
- An important activity that occurs near the end of the project planning phase is the development of the Project Scope Statement. Developed primarily for the customer, this document outlines work that will be done and clearly describes what the project will deliver. The Project Scope Statement is useful to make sure that you, the customer, and other project team members have a clear understanding of the intended project size, duration, and outcomes.

- 10. Setting a Baseline Project Plan
- Once all of the prior project planning activities have been completed, you will be able to develop a Baseline Project Plan. This baseline plan provides an estimate of the project's tasks and resource requirements and is used to guide the next project phase—execution. As new information is acquired during project execution, the baseline plan will continue to be updated.

❑ Executing the Project:

- Project execution puts the Baseline Project Plan into action. Within the context of the SDLC, project execution occurs primarily during the analysis, design, and implementation phases.



- **1. Executing the Baseline Project Plan:**
- This means that you initiate the execution of project activities, acquire and assign resources, orient and train new team members, keep the project on schedule, and ensure the quality of project deliverables. This is a formidable task, but a task made much easier through the use of sound project management techniques.
- **2. Monitoring project progress against the Baseline Project Plan**
- While you execute the Baseline Project Plan, you should monitor your progress. If the project gets ahead of (or behind) schedule, you may have to adjust resources, activities, and budgets. Monitoring project activities can result in modifications to the current plan. Measuring the time and effort expended on each activity will help you improve the accuracy of estimations for future projects. It is possible, with project schedule charts such as Gantt charts, to show progress against a plan, and it is easy with network diagrams to understand the ramifications (the possible results of an action)of delays in an activity. Monitoring progress also means that the team leader must evaluate and appraise (to examine someone or something in order to judge their qualities, success or needs)each team member, occasionally change work assignments or request changes in personnel, and provide feedback to the employee's supervisor.

3. Managing changes to the Baseline Project Plan

- You will encounter pressure to make changes to the baseline plan. Numerous events may initiate a change to the Baseline Project Plan, including the following possibilities:

1. A slipped completion date for an activity
2. A bungled (to do something wrong, in a careless or stupid way) activity that must be redone
3. The identification of a new activity that becomes evident later in the project
4. An unforeseen change in personnel due to sickness, resignation, or termination .

4. Maintaining the project workbook

- As in all project phases, maintaining complete records of all project events is necessary. The workbook provides the documentation new team members require to assimilate (to take in, fit into, or become similar) project tasks quickly. It explains why design decisions were made and is a primary source of information for producing all project reports.

- **5. Communicating the project status**
- The project manager is responsible for keeping all stakeholders— system developers, managers, and customers—abreast (describes two or more people who are next to each other and moving in the same direction) of the project status.

❑ Closing Down the project

- The focus of project closedown is to bring the project to an end. Projects can conclude with a natural or unnatural termination. A natural termination occurs when the requirements of the project have been met—the project has been completed and is a success. An unnatural termination occurs when the project is stopped before completion . Several events can cause an unnatural termination of a project. For example, it may be learned that the assumption used to guide the project proved to be false, that the performance of the systems or development group was somehow inadequate, or that the requirements are no longer relevant or valid in the customer's business environment. The most likely reasons for the unnatural termination of a project relate to running out of time or money, or both.

Representing and Scheduling Project

- A project manager has a wide variety of techniques available for depicting and documenting project plans
- These planning documents can take the form of graphical or textual reports, although graphical reports have become most popular for depicting project plans.
- The most commonly used methods are Gantt charts and network diagrams.
- Gantt charts do not (typically) show how tasks must be ordered (precedence) but simply show when a task should begin and when it should end, they are often more useful for depicting relatively simple projects or subparts of a larger project, showing the activities of a single worker, or monitoring the progress of activities compared to scheduled completion dates.
- • A network diagram shows the ordering of activities by connecting a task to its predecessor and successor tasks.
- Sometimes a network diagram is preferable; other times a Gantt chart more easily shows certain aspects of a project.

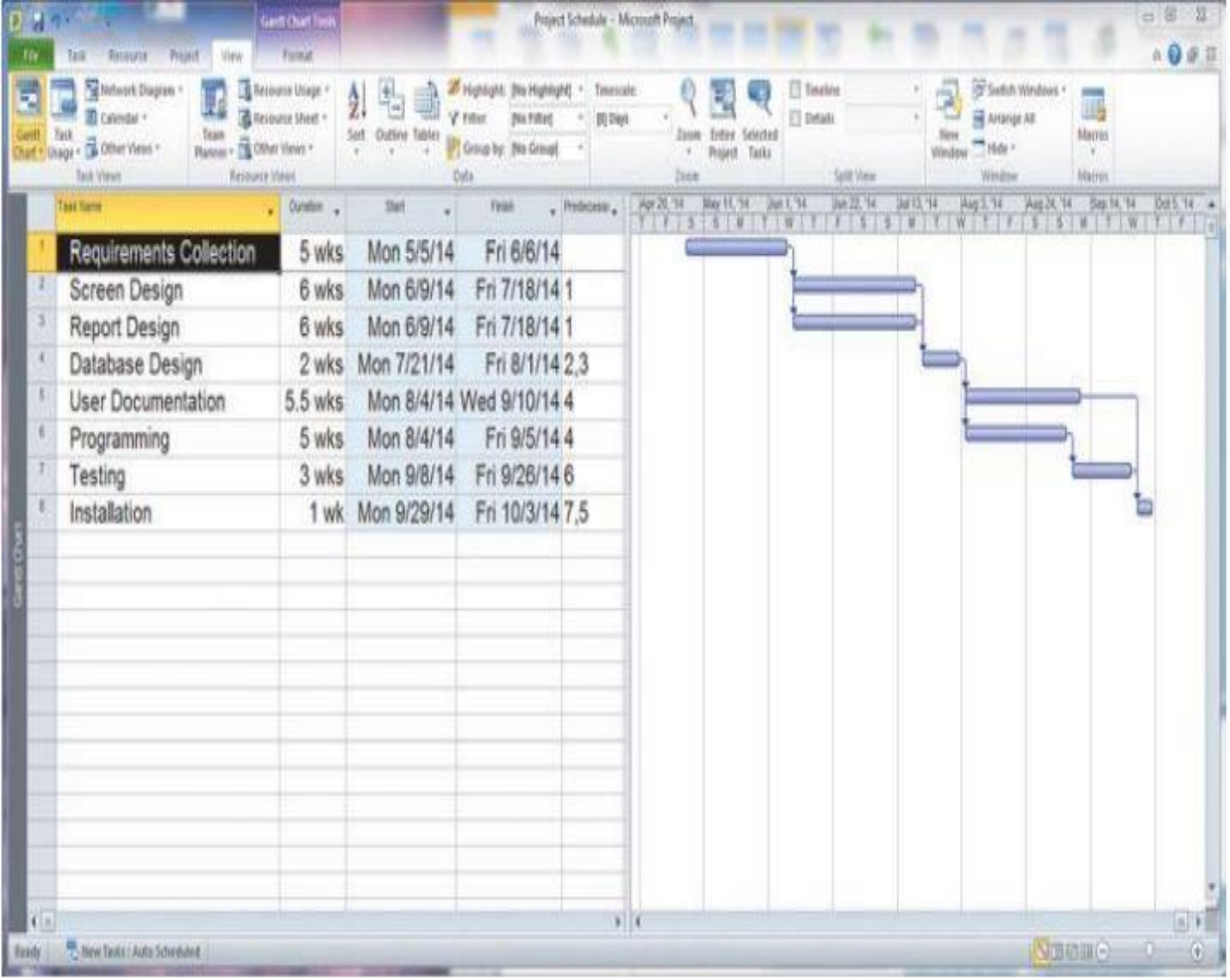


Figure: Graphical diagrams that depict project plans
(a) A Gantt chart

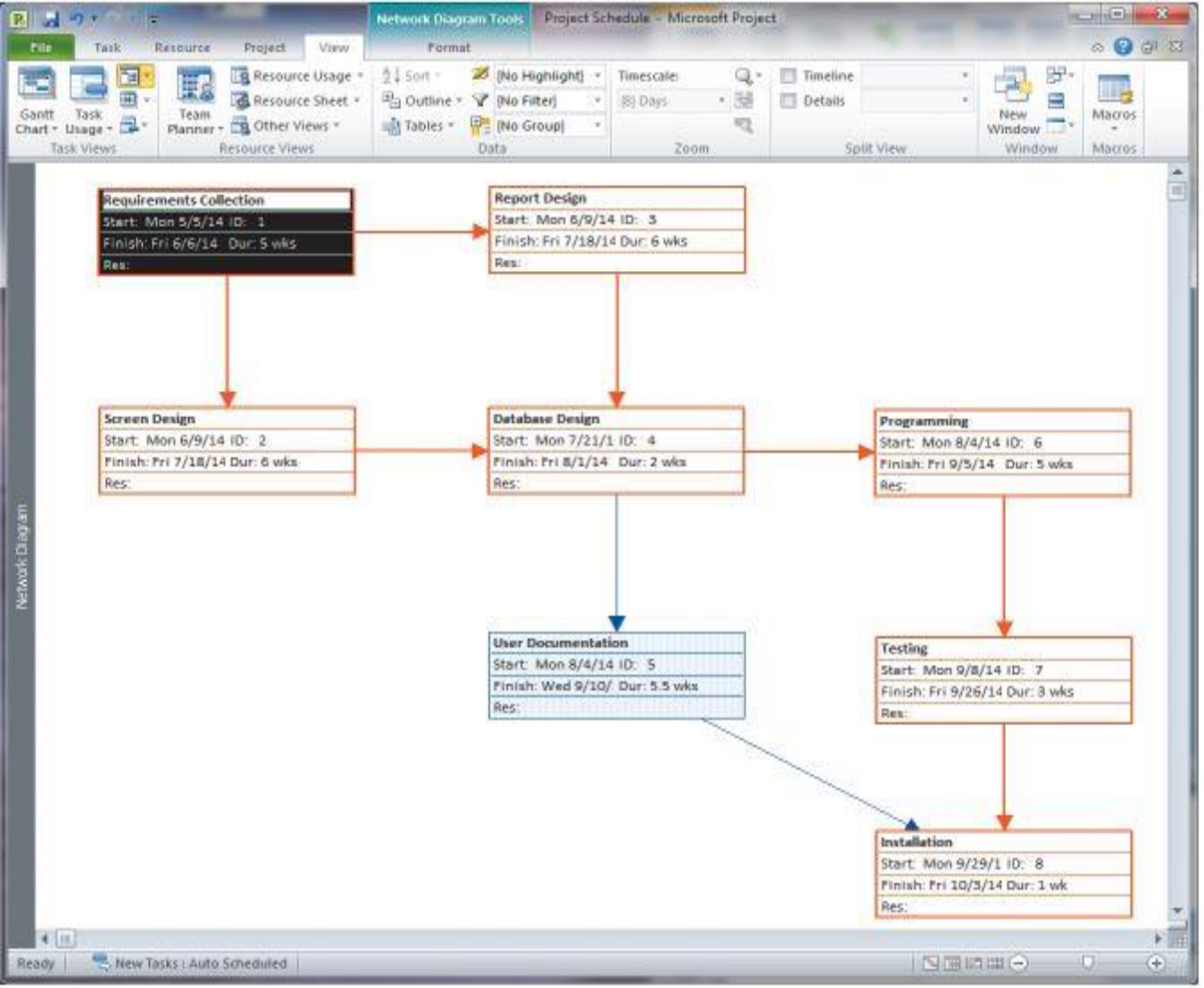


Figure: Graphical diagrams that depict project plans (b) A network diagram

Gantt Charts vs. Network Diagrams

- Here are the key differences between these two charts:
- Gantt charts visually show the duration of tasks, whereas a network diagram visually shows the sequence dependencies between tasks.
- Gantt charts visually show the time overlap of tasks, whereas a network diagram does not show time overlap but does show which tasks could be done in parallel.
- • Some forms of Gantt charts can visually show slack time available within an earliest start and latest finish duration. A network diagram shows this by data within activity rectangles.

Representing Project Plans:

- Project scheduling and management require that time, costs, and resources be controlled. Resources are any person, group of people, piece of equipment, or material used in accomplishing an activity
- Network diagramming is a critical path scheduling technique used for controlling resources.
- A critical path refers to a sequence of task activities whose order and durations directly affect the completion date of a project.
- A network diagram is one of the most widely used and best known scheduling methods.
- You would use a network diagram when tasks:

1. are well defined and have a clear beginning and end point,
2. can be worked on independently of other tasks,
3. are ordered, and
4. serve the purpose of the project

- A major strength of network diagramming is its ability to represent how completion times vary for activities.
- Because of this, it is more often used than Gantt charts to manage projects such as information systems development, where variability in the duration of activities is the norm.
- Network diagrams are composed of circles or rectangles representing activities and connecting arrows showing required work flows

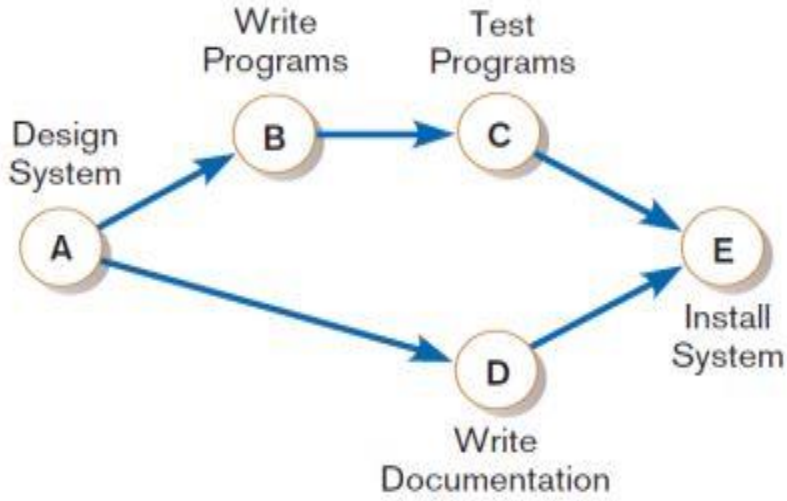


Figure
A network diagram showing activities (represented by circles) and sequence of those activities (represented by arrows)

Calculating Expected time durations using PERT:(Program Evaluation Review Technique)

- One of the most difficult and most error-prone activities when constructing a project schedule is the determination of the time duration for each task within a work breakdown structure. It is particularly problematic to make these estimates when there is a high degree of complexity and uncertainty about a task. PERT is a technique that uses optimistic, pessimistic, and realistic time estimates to calculate the expected time for a particular task. This technique can help you to obtain a better time estimate when there is some uncertainty as to how much time a task will require to be completed. The optimistic (o) and pessimistic (p) times reflect the minimum and maximum possible periods of time for an activity to be completed. The realistic (r) time, or most likely time, reflects the project manager's "best guess" of the amount of time the activity actually will require for completion. Once each of these estimates is made for an activity, an expected time (ET) can be calculated. Because the expected completion time should be closest to the realistic (r) time, it is typically weighted four times more than the optimistic (o) and pessimistic (p) times. Once you add these values together, it must be divided by six to determine the ET. This equation is shown in the following formula:

$$ET = \frac{o + 4r + p}{6}$$

- where
- ET = expected time for the completion for an activity
- o = optimistic completion time for an activity
- r = realistic completion time for an activity
- p = pessimistic completion time for an activity

- For example, suppose that your instructor asked you to calculate an expected time for the completion of an upcoming programming assignment. For this assignment, you estimate an optimistic time of two hours, a pessimistic time of eight hours, and a most likely time of six hours. Using PERT, the expected time for completing this assignment is 5.67 hours

Using project management software

- A wide variety of automated project management tools is available to help you manage a development project.
- New versions of these tools are continuously being developed and released by software vendors.
- Most of the available tools have a set of common features that include the ability to define and order tasks, assign resources to tasks, and easily modify tasks and resources.
- Project management tools are available to run on IBM-compatible personal computers, the Macintosh, and larger mainframe and workstation-based systems.
- These systems vary in the number of task activities supported, the complexity of relationships, system processing and storage requirements, and, of course, cost.
- When using this system to manage a project, you need to perform at least the following activities:

1. Establish a project starting or ending date.
2. Enter tasks and assign task relationships.
3. Select a scheduling method to review project reports.

- Project management software helps project managers (PMs) and teams collaborate and meet goals on time while managing resources and cost. Functions may include task distribution, time tracking, budgeting, resource planning, team collaboration, and many more
- People also refer to project management software as Task Management Software or Project Portfolio Management (PPM).
- Project management software covers a range of platforms, each with a slightly different mix of functionality. It's crucial that the vendor you select makes your projects easier to manage and doesn't add unneeded complexity. The transition should be as smooth as possible.
- The three major pillars of project management are planning, tracking, and collaboration.

Project Start Date

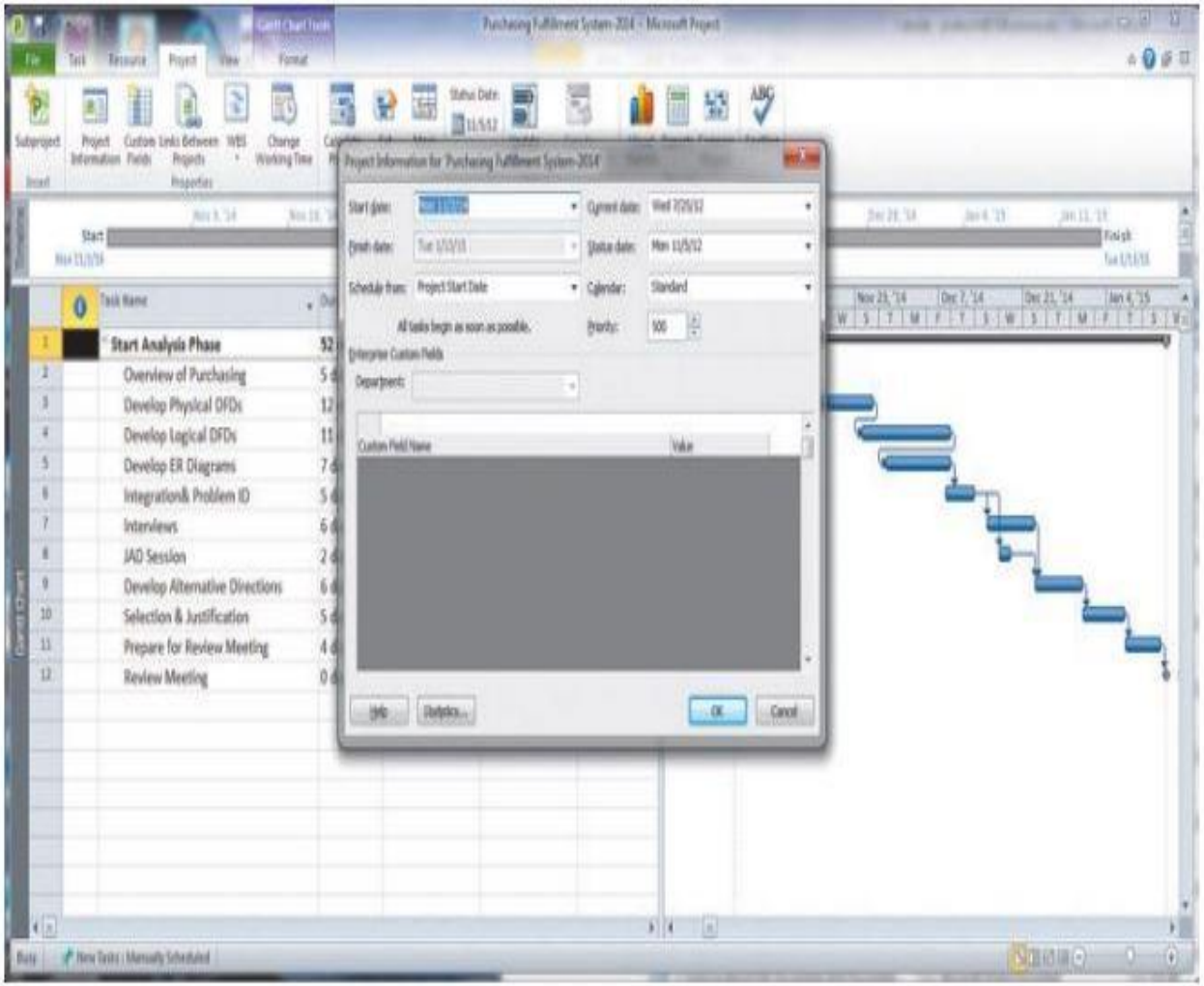


FIGURE
Establishing a project starting date in Microsoft Project for Windows (*Source:* Microsoft Corporation.)

Entering Tasks

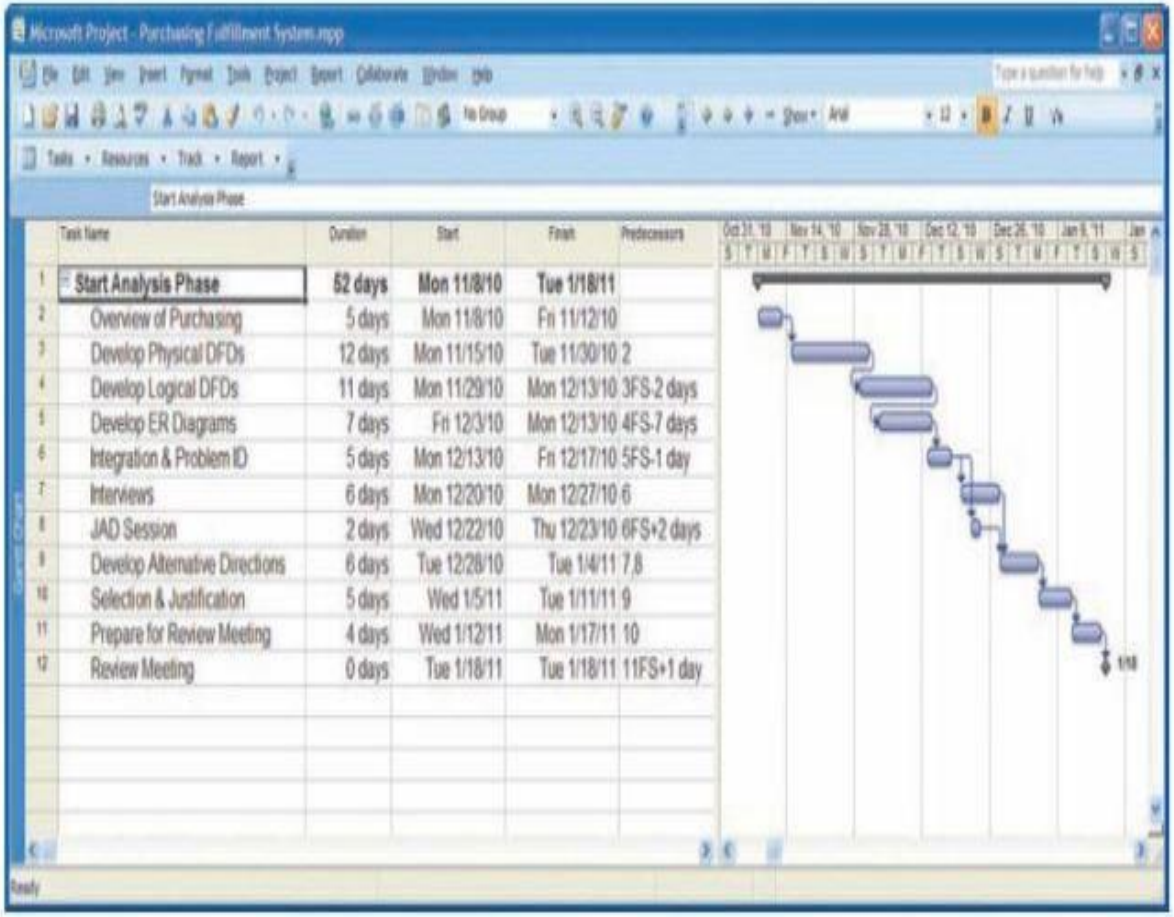


FIGURE
Entering tasks and assigning task relationships in Microsoft project for Windows (*Source:* Microsoft Corporation.)

Viewing Network Diagram

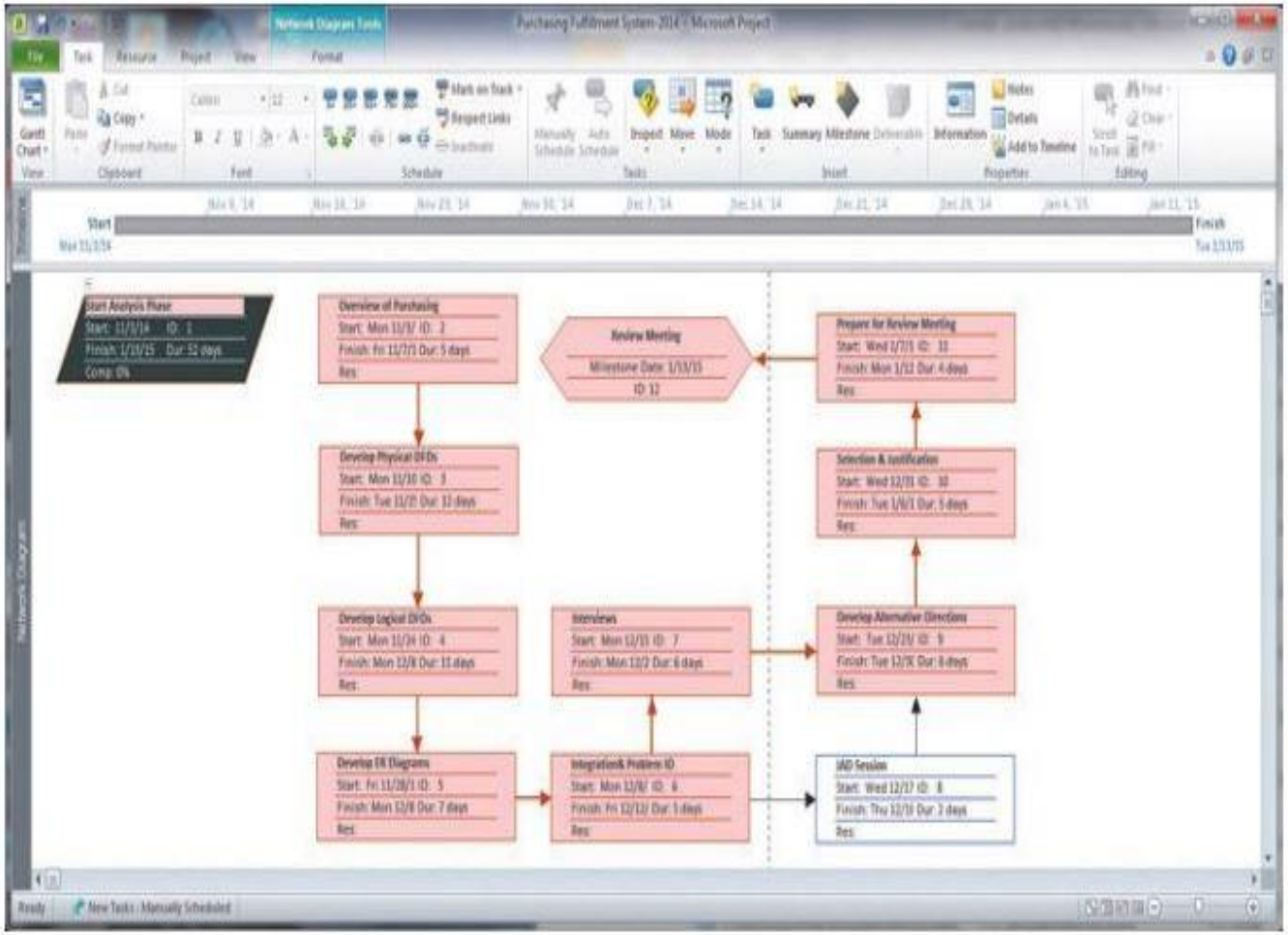
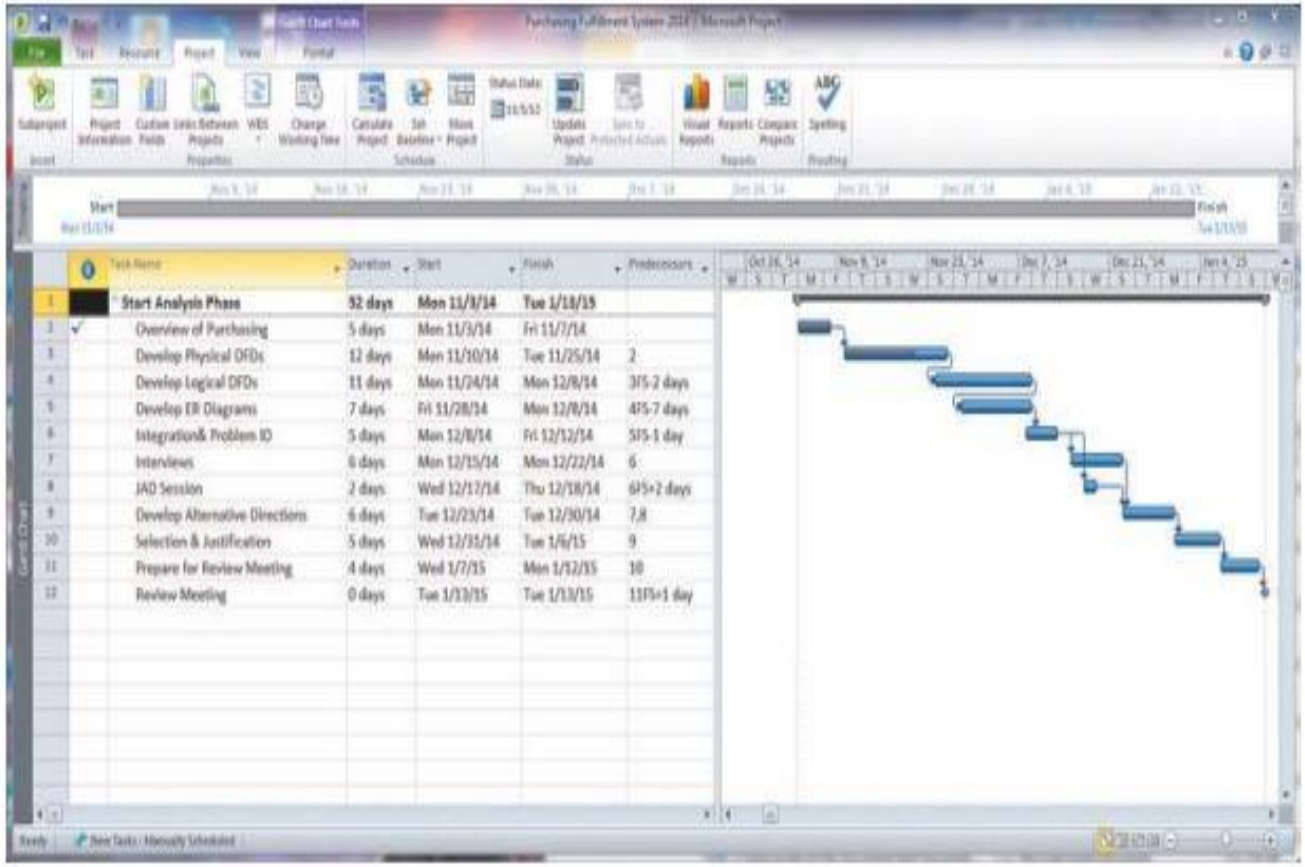


FIGURE: Viewing project information as a network diagram in Microsoft Project for Windows (*Source:* Microsoft Corporation.)

Hexagon shape indicates a milestone.

Red boxes and arrows indicate critical path (no slack).

Viewing Gantt Chart



FIGURE

Gantt chart showing progress of activities (right frame) versus planned activities (left frame)

Black line at top indicates a summary activity (composed of subtasks).
Diamond shape indicates a milestone.

Assignment-1:

- What is an Information System? Explain its types.
- Explain System Development Life Cycle in Details.
- Explain Agile method with its merits and demerits.
- Explain the roles of System Analyst in detail.
- Discuss the waterfall model with its merits and demerits.
- Explain CASE tools and its application.


Thank
you