Course Name	**BIT 312 - Information Management**	
Credit Units	3	
Pre-requisite	Information Technology Fundamentals	
Purpose of the Course	To prepare student to understand information system management, to identify organizational situations that can be supported by computerized management information systems (MIS), and to provide hands-on experience on designing and implementing an automated system	
Expected Learning Outcomes	On completion of this module the students will be able to: a) Explain the role of data, information, database management systems and data mining in organizations appreciating the advantages of a database approach compared to traditional file processing. b) Identify and explain the general types of databases: personal, workgroup, department, enterprise. c) Describe mechanisms for data collection and their implications . d) Explain basic issues of data retention, including the need for retention, physical storage, security. e) Explain why data backup is important and how organizations use backup and recovery systems.	
Course Content	Information systems: purpose, use, value; Properties of data (quality, accuracy, timeliness); Database systems, Analysis of data, forms and sources; Data collection; Data retention; Information backup and recovery	
Mode of delivery	Lectures, directed reading, practical demonstrations of communication services, and hands-on laboratory sessions and projects.	
Instructional Material and/or Equipment	Audio Visual Equipment, Whiteboard, computer simulation software, computer programming tools	
Course Assessment	Type	Weighting (%)
	Examination	70
	Continuous Assessment	30
	Total	100
Core Reading Material for the course	Management of Information Security, Herbert Mattord, Michael Whitman, 2004, Course Technology	
Recommended reference material	ACM Transactions on Information and System Security (TISSEC) Learning and Using Geographic Information Systems: ArcExplorer Edition Wilpen Gorr, Kristen Kurland, 2005, Course Technology James C. Brancheau, Brian D. Janz, James C. Wetherbe, “	

COURSE CODE: BIT 312
COURSE TITLE: INFORMATION MANAGEMENT
COURSE OBJECTIVES:
- To understand the importance of information in business
- To know the techniques and methods used for effective decision making
- To understand the security control and ethics in IT

COURSE OUTLINE

TOPIC 1: INTRODUCTION
Data, Information, Intelligence, Information Technology, Information System, evolution, system development methodologies, functional information systems, DSS, EIS, information system.

TOPIC 2: SYSTEM ANALYSIS AND DESIGN
System flow chart, Decision table, Data flow diagram (DFD), entity relationship ER, object oriented analysis and design, UML diagram.

TOPIC 3: DATABASE MANAGEMENT SYSTEMS
DBMS-HDBMS, NDBMS, RDBMS, OODBMS, querry processing, Data mart

TOPIC 4: SECURITY CONTROL AND REPORTING
Security, testing, error detection, controls, IS vulnerability, Disaster management, computer crimes, security, securing the web, intranets and wireless networks, software audit, ethics in IT, user interface and reporting

TOPIC 5: NEW INITIATIVES
Role of information Management in computing, cloud computing, CMM

COURSE OUTCOMES
On completion of the course, student will be able to
-Understand the importance of information in business
-Gains knowledge on effective applications of information systems in business
-Describe about the system analysis and designing
-Manage the database management system.
-Develop the security control to protect the system
-Know the role of information management in e-business, ERP and business intelligence

TOPIC 1 INTRODUCTION
Data, Information, Intelligence, Information Technology, Information System, Evolution, Types based on functions and Hierarchy, System Development Methodologies, Functional Information Systems, DSS, EIS, KMS, GIS, International Information System.

Data:
Data are facts that are not currently being used for decision making purpose. For example, payroll records, data on accounts receivable, personnel data etc.

“Data refers to a collection of facts usually collected as the result of experience, observation or experiment, or processes within a computer system, or a set of premises.”
Data may consist of numbers, words, or images, particularly as measurements

or observations of a set of variables. Data is often viewed as a lowest level of abstraction from which information and knowledge are derived.

Information

It is data that has been given meaning by way of relational connection. This "meaning" can be useful, but does not have to be. In computer parlance, a relational database makes information from the data stored within it.
Information is derived from data and useful in solving problems. Information, therefore is a potential function of data.
Once the data have been sifted through and organized so as to be relevant to the context of the decision at hand, the data can be called information.

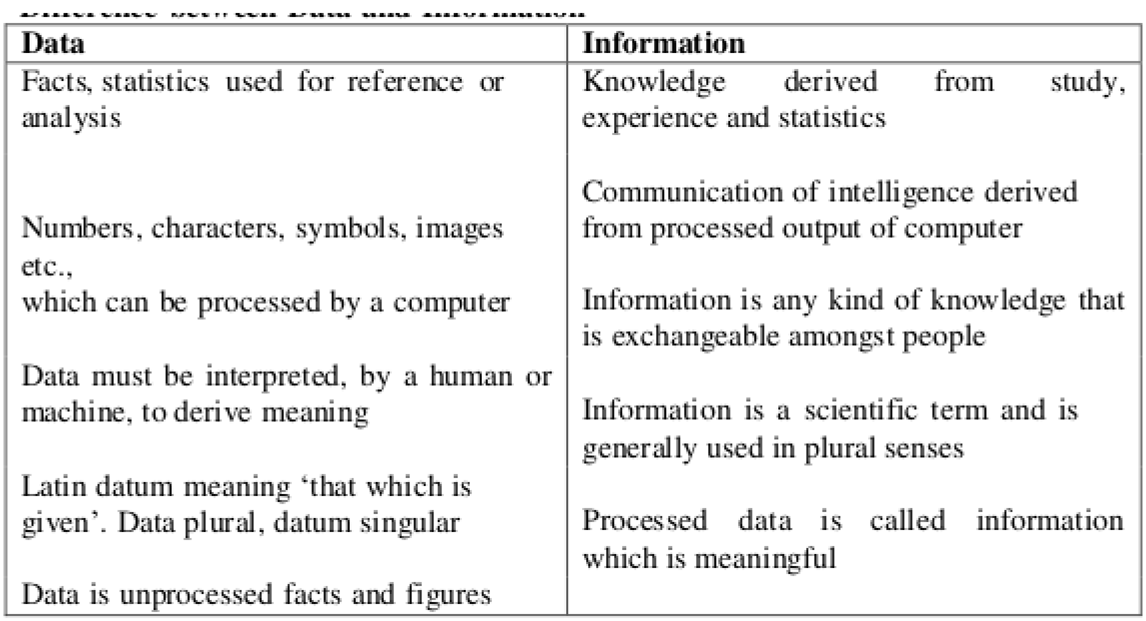
Important points regarding information

- ✓ Organized form of data
- ✓ Data with context
- ✓ Data with relationships
- ✓ Pies of knowledge that can be codified and stored
- ✓ Physical representation of knowledge
- ✓ Pooling of bits of knowledge
- ✓ Communication or reception of knowledge
- ✓ Action of informing

Characteristics of Information

- ✓ Timeliness
- ✓ Accuracy
- ✓ Frequency
- ✓ Relevant
- ✓ Appropriateness
- ✓ Conciseness
- ✓ Understandability
- ✓ Complete
- ✓ Current
- ✓ Economica

Difference between Data and Information



Data	Information
Facts, statistics used for reference or analysis	Knowledge derived from study, experience and statistics
Numbers, characters, symbols, images etc., which can be processed by a computer	Communication of intelligence derived from processed output of computer
	Information is any kind of knowledge that is exchangeable amongst people
Data must be interpreted, by a human or machine, to derive meaning	Information is a scientific term and is generally used in plural senses
Latin datum meaning 'that which is given'. Data plural, datum singular	Processed data is called information which is meaningful
Data is unprocessed facts and figures	

Intelligence:

Intelligence is related to the possession and creation of knowledge and characterizes an adaptive behavior.

Intelligence is a way of manipulating information and knowledge to get certain things done, one of which is the acquiring of more information and knowledge

Knowledge

Knowledge is the facts, feelings or experiences known by a person or a group of people. It is considered to be present in ideas, judgment, root causes, relationships, perspectives and concepts. Knowledge is the result of learning and is stored in an individual brain or encoded in documents, product, facilities and concepts. It is gained from the certitude that comes from experience, a logical argument, or a preponderance of evidence. Knowledge knows what works and how it works.

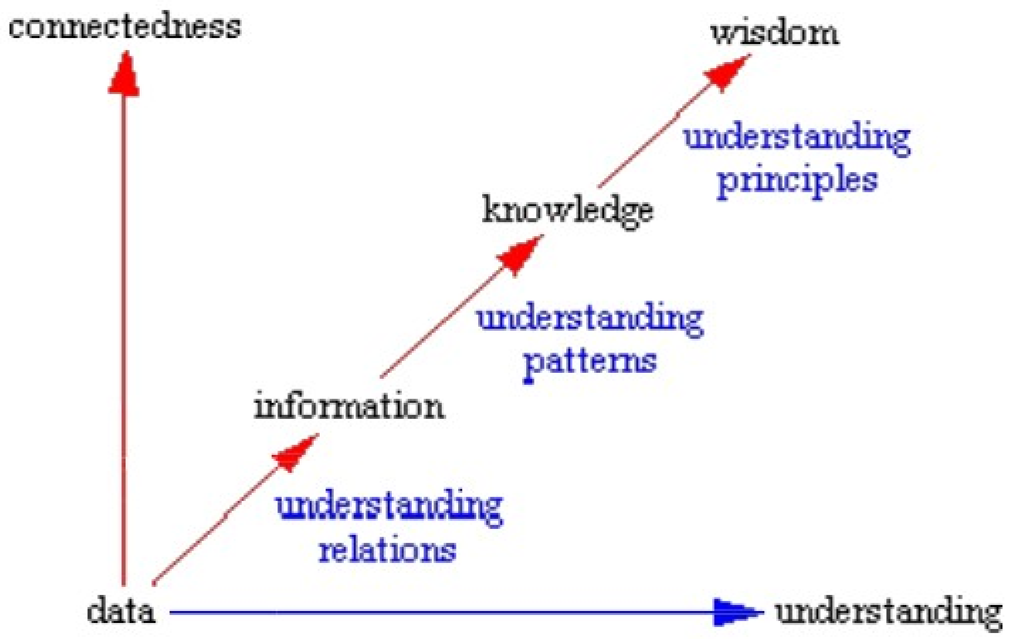
Understanding

It is cognitive and analytical. It is the process by which I can take knowledge and synthesize new knowledge from the previously held knowledge. The difference between understanding and knowledge is the difference between "learning" and "memorizing". People who have understanding can undertake useful actions because they can synthesize new knowledge, or in some cases, at least new information, from what is previously known (and understood). That is, understanding can build upon currently held information, knowledge and understanding itself. In computer parlance, AI systems possess understanding in the sense that they are able to synthesize new knowledge from previously stored information and knowledge.

Wisdom

Wisdom is an extrapolative and non-deterministic, non-probabilistic process. It calls upon all the previous levels of consciousness, and specifically upon special types of human programming (moral, ethical codes, etc.). It beckons to give us understanding about which there has previously been no understanding, and in doing so, goes far beyond understanding itself. It is the essence of philosophical probing. Unlike the previous four levels, it asks questions to which there is no (easily- achievable) answer, and in some cases, to which there can be no humanly-known answers period. Wisdom is therefore, the process by which we also discern, or judge, between right and wrong, good and bad.

Relationship between Data, Information, Knowledge and Intelligence



According to Russell Ackoff, a systems theorist and professor of organizational change, the content of the human mind can be classified into five categories:
1. **Data**: symbols
2. Information: data that are processed to be useful; provides answers to "who", "what", "where", and "when" questions
3. Knowledge: application of data and information; answers "how" questions
4. Understanding: appreciation of "why"
5. Wisdom: evaluated understanding.

Information Technology
Everyone is using Information Technology knowingly or unknowingly. It is growing rapidly. It covers many different and distinct fields like movies, wireless phones or internet. IT is used everywhere in any field.
User can use IT, for creating multimedia in Business or for creating different magazine or websites. It not only helps the organization or to the societies but it is finding helpful to the individual persons. In past days, everything is manual processing for mailing system; preparing reports were carried without electronic machine.

Definition: "IT enhances our local economy. It helps to solving social issue and to develop community relationship. Use of IT tools is equitable and affordable". Information technology is a system that process for required output. IT can be described as a set of elements connect together for retrieving, processing and outputting data in the appropriate format for the purpose of common objective.

Information technology is a concept. It is not a hardware part or a device. It is a technology that helps to process information. The information system which is used to help to build information technology is called information system. Information system is a collection of hardware and software. It helps to perform different function.

Components of IT

- ✓ People
- ✓ Hardware
- ✓ Software
- ✓ Data
- ✓ Network

Advantages of IT

1. Globalization
2. Cost effectiveness
3. All time
4. Communication
5. Bridging cultural gap
6. Creation of new jobs

Disadvantages of IT

- ✓ Unemployment
- ✓ Privacy and safety issues
- ✓ Lack of job security
- ✓ Dominant culture
- ✓ Literary
- ✓ Backup
- ✓ Affect human relationship

Applications of IT

1. Education
- Access to variety of learning resources
- Any time learning
- Collaborative learning

2. Business and Industry
- Customer relations
- Business operations
- Industrial productivity
- Business mobility

3. Entertainment
- Games,
- Music and videos,
- Animated movies,
- CDMA or GPRS etc.

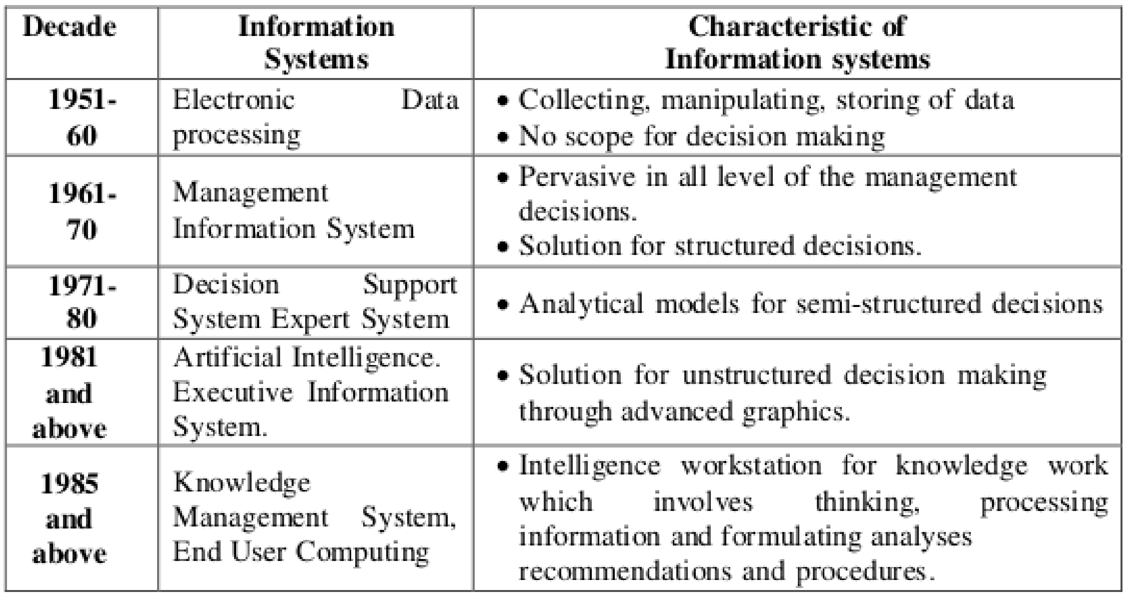
4. Medicine
- Magnetic resonance imaging (MRI)
- Computerized axial tomography (CAT)
- 5. Robotic surgery etc.

Challenges of IT

- ✓ Challenge of globalization
- ✓ Challenge of insularity
- ✓ Challenge of privary
- ✓ Challenges of ethics

Information System (IS)
IS is a set of people, procedures and resources that collects, transforms, and disseminates information in an organization
“An information system (IS) can be any organized combination of people, hardware, software, communications networks, and data resources that stores and retrieves, transforms, and disseminate information in an organization”.
Information systems are implemented within an organization for the purpose of improving the effectiveness and efficiency of that organization.

Evolution of Management Information System



Decade	Information Systems	Characteristic of Information systems
1951-60	Electronic Data processing	• Collecting, manipulating, storing of data • No scope for decision making
1961-70	Management Information System	• Pervasive in all level of the management decisions. • Solution for structured decisions.
1971-80	Decision Support System Expert System	• Analytical models for semi-structured decisions
1981 and above	Artificial Intelligence. Executive Information System.	• Solution for unstructured decision making through advanced graphics.
1985 and above	Knowledge Management System, End User Computing	• Intelligence workstation for knowledge work which involves thinking, processing information and formulating analyses recommendations and procedures.

Types of Information Systems:
- ✓ Information system based on functions
- ✓ Information system based on management hierarchy

Information systems based on functions
- ✓ Manufacturing
- ✓ Accounting
- ✓ Finance
- ✓ Marketing
- ✓ Sales
- ✓ Human resource

Information systems based on management hierarchy
- ✓ Executive information systems
- ✓ Decision support systems
- ✓ Management information systems
- ✓ Transaction processing system

IS Activities

- ✓ Input
- ✓ Process
- ✓ Output
- ✓ Storage
- ✓ Control

Capabilities of information systems

- ✓ Provide fast and accurate transaction processing
- ✓ Provide large capacity, fast access storage
- ✓ Provide fast communication
- ✓ Reduce information overload
- ✓ Span boundaries
- ✓ Provide support for decision making
- ✓ Provide a competitive weapon

System Development Methodologies

A methodology is a system of methods, or a body of methods, rules and postulates used by a discipline. So, a systems methodology is a body of methods, rules and postulates used by a system practitioners to investigate, understand and address systems, their issues, problems, behaviors and contexts, and – where appropriate – to moderate, modify, or otherwise address and solve, resolve, or dissolve issues and problems.

Objectives of system methodology

- ✓ It shows how the system should be working
- ✓ It examines how various components work together to produce a particular outcome
- ✓ It shows the processes as part of a larger system
- ✓ It reveals data collection needs
- ✓ It can be helpful in monitoring performance

System Development:

Systems development has two major components:

- ✓ System Analysis and
- ✓ Systems Design

System Analysis: It is the process of gathering and interpreting facts, diagnosing problems, and using the information to recommend improvements to the system.

System Design: It is the process of planning a new business system or complement an existing system.

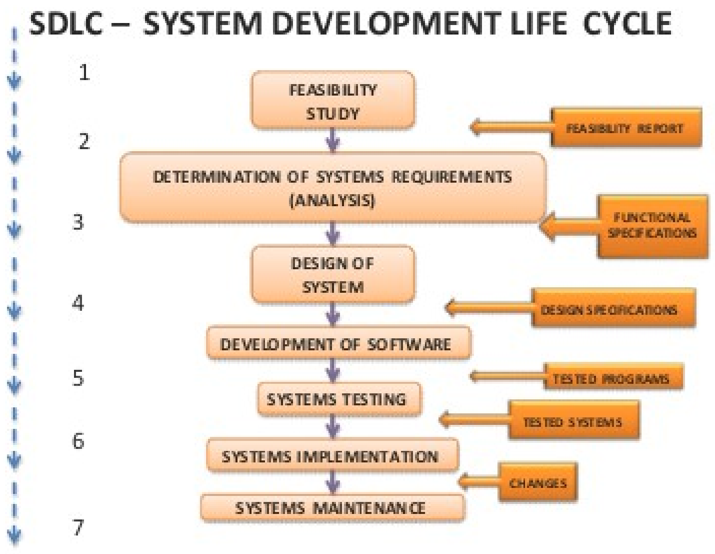
Analysis specifies "what" the system should do, that is it sets the objective and Design states "how" to accomplish this objective.

Systems Development Life Cycle (SDLC)

The Systems development life cycle is a sequence of events carried out by analysts, designers and users to develop and implement an information system. These activities are carried out in different stages.

Phases of SDLC:

1. Preliminary investigation / Feasibility Study
2. Determination of Systems requirements
3. Design of the system
4. Development of software
5. Systems testing
6. Systems implementation
7. Systems Maintenance



1. Preliminary Investigation
One must know what the problem is before it can be solved. An important outcome of the preliminary investigation is determining whether the system requested is feasible or not. The major purposes of this study are:

- ✓ Identify the responsible users and develop an initial scope of the system
- ✓ Identify current deficiencies in the user's environment
- ✓ Determine objectives for the new system
- ✓ Determine whether it is feasible to automate the system and, if so, suggest some acceptable options

The three major areas to consider while determining the feasibility are:
a**) Technical Feasibility:** The analyst must find out whether current technical resources which are available in the organization is capable of handling the user's requirements. If not, then the analyst with the help of vendors should confirm whether the technology is available and capable of meeting the user's request.

b) Economic Feasibility:

- ✓ Management time
- ✓ Time spent by the systems analysis team
- ✓ Cost of doing the full systems study
- ✓ Estimated cost of hardware
- ✓ Estimated cost of software and / or software development

c) Operational Feasibility:

- ✓ Operational feasibility is dependent upon determining human resources for the project
- ✓ If the ultimate users are virtually wedded to the present system and they see no problem and if they are not involved in requesting for a new system, then resistance to its operation will be strong.
- ✓ Alternatively, if users themselves have expressed a need for an improved system, then they will put in all efforts to see that it becomes operational, and will eventually use it.

2. Determination of Requirements (Analysis)
This activity may be carried out in two phases:
a. Detailed Investigation:
- ✓ What is being done by the current system?
- ✓ How it is being done?
- ✓ How frequently does it occur?
- ✓ How big is the volume of transactions or decisions?
- ✓ How well is the task being performed?
- ✓ Does a problem exist? If so, How serious it is?

b. Analysis or Determination of System Requirement:
- ✓ Inputs must be received by the system
- ✓ Output must be produced by the system
- ✓ Data to be retained
- ✓ The procedures to get the output from the given input
- ✓ Audit & control requirements
- ✓ System Acceptance Criteria

3. Design of the System
The Design process should take care of the following:
- ✓ Identification of reports and outputs
- ✓ Scrutinize the data present on each report/output
- ✓ Sketch the form as expected to appear at the end of completion of the system
- ✓ Description of data to be input calculated/stored
- ✓ Individual data items and calculation procedures

4. Development of Software
In this stage, the actual coding/writing of the programs is done. In some firms, separate groups of programmers do the programming whereas other firms employ analyst-programmers who do analysis and design as well as code programs.
Programmers are also responsible for documenting the program including comments that explain both how and why a certain procedure was coded in a specific way.

5. Systems Testing
- ✓ Once the programs are tested individually, then the system as a whole needs to be tested.
- ✓ During testing, the system is used experimentally to ensure that the software does not fail.
- ✓ Special test data is prepared as input for processing and the results are examined to locate unexpected results.
- ✓ In many organizations testing is performed by persons other than who wrote the original programs.
- ✓ Using persons who do not know how the programs were designed ensures more complete and reliable software

6. Systems Implementation
- ✓ In this stage, the systems analysts put the new software which has been tested into use.

- ✓ User personnel are trained and any files of data needed by the new system are constructed.
- ✓ In short, the new software is installed and then used.

7. Systems Maintenance

- ✓ Once installed, the software is often used for many years.
- ✓ However, both the organization and the users change.
- ✓ The environment may also change over a period of time.
- ✓ Therefore the software has to be maintained and changes will be made to the software, files or procedures to meet the user's requirements.

Strength of the SDLC

- ✓ Well tried and tested
- ✓ Provides a base guideline for systems development which can be modified to suit specific requirements
- ✓ Emphasis on project control, documentation, standards, and quality control.
- ✓ Useful for building large transaction processing systems and Management Information Systems
- ✓ Building complex systems which need rigorous and formal requirements analysis, and tight control of the system development process.

Limitations of the SDLC

- ✓ It is resource intensive
- ✓ It is inflexible and inhibits change
- ✓ Something that may no longer be appropriate
- ✓ Top down, approach discourages iteration
- ✓ Hard to visualize final system
- ✓ Not well suited to most of small desktop systems
- ✓ Does not encourage user participation
- ✓ Focus on technical aspects
- ✓ Costly and time consuming
- ✓ Inflexible to discourage change

System / Software Life Cycle Model:

Definition:

A (software/system) lifecycle model is a description of the sequence of activities carried out in a project, and the relative order of these activities.

It provides a fixed generic framework that can be tailored to a specific project.

Project specific parameters will include:

- Size, (person-years)
- Budget,
- Duration.

Project Plan = Lifecycle Model + Project Parameters

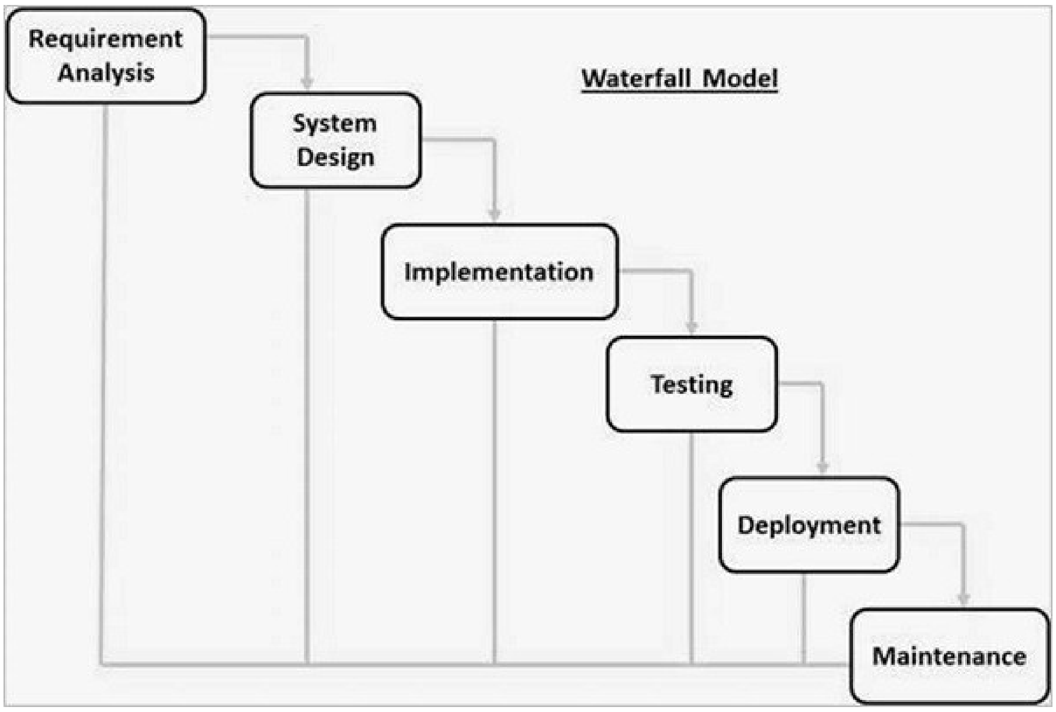
There are hundreds of different lifecycle models to choose from, e.g:

- Waterfall Model
- Iterative model
- Spiral Model

• Evolutionary Model
• Prototyping Model
but many are minor variations on a smaller number of basic models.

By changing the lifecycle model, we can improve :
• Development speed (time to market)
• Product quality
• Project visibility
• Administrative overhead
• Risk exposure
• Customer relations, etc.

The Waterfall Model/linear-sequential life cycle model
• The waterfall model is the classic lifecycle model – it is widely known, understood and (commonly) used. Each phase must be completed before the next phase can begin and there is no overlapping in the phases.
• Introduced by Royce 1970.



Requirement Gathering and analysis − All possible requirements of the system to be developed are captured in this phase and documented in a requirement specification document.

•**System Design** − The requirement specifications from first phase are studied in this phase and the system design is prepared. This system design helps in specifying hardware and system requirements and helps in defining the overall system architecture.
•**Implementation** − With inputs from the system design, the system is first developed in small programs called units, which are integrated in the next phase. Each unit is developed and tested for its functionality, which is referred to as Unit Testing.

•**Integration and Testing** − All the units developed in the implementation phase are integrated into a system after testing of each unit. Post integration the entire system is tested for any faults and failures.
•**Deployment of system** − Once the functional and non-functional testing is done; the product is deployed in the customer environment or released into the market.
•**Maintenance** − There are some issues which come up in the client environment. To fix those issues, patches are released. Also to enhance the product some better versions are released. Maintenance is done to deliver these changes in the customer environment.

Advantages

- ✓ Easy to understand and implement.
- ✓ Widely used and known
- ✓ Reinforces good habits: define-before- design, design-before-code
- ✓ Identifies deliverables and milestones
- ✓ Document driven
- ✓ Works well on mature products and weak teams

Disadvantages

1. Idealised, doesn't match reality well.
2. Doesn't reflect iterative nature of exploratory development.
3. Unrealistic to expect accurate requirements so early in project
4. Software is delivered late in project, delays discovery of serious errors.
5. Difficult to integrate risk management
6. Difficult and expensive to make changes to documents, "swimming upstream".
7. Significant administrative overhead, costly for small teams and projects.

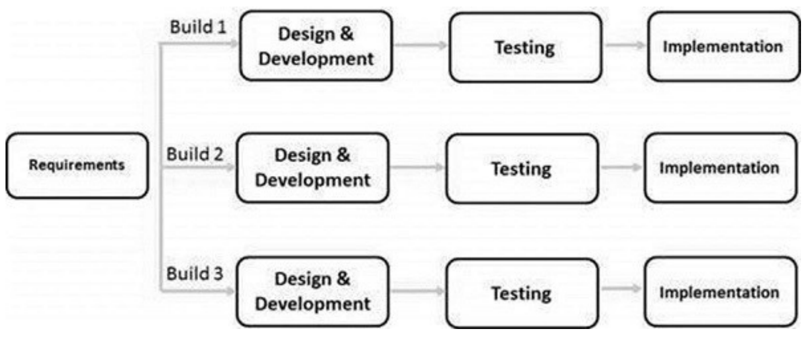
When to use Waterfall Model

• Requirements are very well known
• Product definition is stable
• Technology is understood
• New version of an existing product
• Porting an existing product to a new platform.

Iterative Model - Design

Iterative process starts with a simple implementation of a subset of the software requirements and iteratively enhances the evolving versions until the full system is implemented. At each iteration, design modifications are made and new functional capabilities are added. The basic idea behind this method is to develop a system through repeated cycles (iterative) and in smaller portions at a time (incremental).

The following illustration is a representation of the Iterative and Incremental model −



Iterative and Incremental development is a combination of both iterative design or iterative method and incremental build model for development. "During software development, more than one iteration of the software development cycle may be in progress at the same time." This process may be described as an "evolutionary acquisition" or "incremental build" approach."

In this incremental model, the whole requirement is divided into various builds. During each iteration, the development module goes through the requirements, design, implementation and testing phases. Each subsequent release of the module adds function to the previous release. The process continues till the complete system is ready as per the requirement.

The key to a successful use of an iterative software development lifecycle is rigorous validation of requirements, and verification & testing of each version of the software against those requirements within each cycle of the model. As the software evolves through successive cycles, tests must be repeated and extended to verify each version of the software.

Iterative Model - Application

Like other SDLC models, Iterative and incremental development has some specific applications in the software industry. This model is most often used in the following scenarios −

- Requirements of the complete system are clearly defined and understood.
- Major requirements must be defined; however, some functionalities or requested enhancements may evolve with time.
- There is a time to the market constraint.
- A new technology is being used and is being learnt by the development team while working on the project.
- Resources with needed skill sets are not available and are planned to be used on contract basis for specific iterations.
- There are some high-risk features and goals which may change in the future.

The **advantages** of the Iterative and Incremental SDLC Model are as follows −

- Some working functionality can be developed quickly and early in the life cycle.
- Results are obtained early and periodically.
- Parallel development can be planned.
- Progress can be measured.
- Less costly to change the scope/requirements.
- Testing and debugging during smaller iteration is easy.
- Risks are identified and resolved during iteration; and each iteration is an easily managed milestone.
- Easier to manage risk - High risk part is done first.
- With every increment, operational product is delivered.
- Issues, challenges and risks identified from each increment can be utilized/applied to the next increment.
- Risk analysis is better.

•It supports changing requirements.
•Initial Operating time is less.
•Better suited for large and mission-critical projects.
•During the life cycle, software is produced early which facilitates customer evaluation and feedback.

The **disadvantages** of the Iterative and Incremental SDLC Model are as follows −

•More resources may be required.
•Although cost of change is lesser, but it is not very suitable for changing requirements.
•More management attention is required.
•System architecture or design issues may arise because not all requirements are gathered in the beginning of the entire life cycle.
•Defining increments may require definition of the complete system.
•Not suitable for smaller projects.
•Management complexity is more.
•End of project may not be known which is a risk.
•Highly skilled resources are required for risk analysis.
•Projects progress is highly dependent upon the risk analysis phase.

The Spiral Model
Since end-user requirements are hard to obtain/define, it is natural to develop software in an experimental way:
e.g.
1. Build some software
2. See if it meets customer requirements
3. If no go to 1 else stop.
This loop approach gives rise to structured iterative lifecycle models. In 1988 Boehm developed the spiral model as an iterative model which includes risk analysis and risk management.
The spiral model has four phases. A software project repeatedly passes through these phases in iterations called Spirals.

Identification

This phase starts with gathering the business requirements in the baseline spiral. In the subsequent spirals as the product matures, identification of system requirements, subsystem requirements and unit requirements are all done in this phase.

This phase also includes understanding the system requirements by continuous communication between the customer and the system analyst. At the end of the spiral, the product is deployed in the identified market.

Design

The Design phase starts with the conceptual design in the baseline spiral and involves architectural design, logical design of modules, physical product design and the final design in the subsequent spirals.

Construct or Build

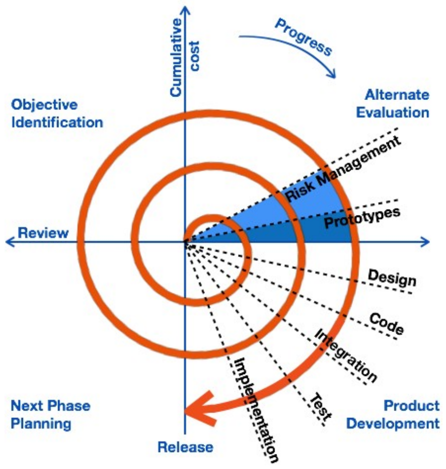
The Construct phase refers to production of the actual software product at every spiral. In the baseline spiral, when the product is just thought of and the design is being developed a POC (Proof of Concept) is developed in this phase to get customer feedback.

Then in the subsequent spirals with higher clarity on requirements and design details a working model of the software called build is produced with a version number. These builds are sent to the customer for feedback.

Evaluation and Risk Analysis

Risk Analysis includes identifying, estimating and monitoring the technical feasibility and management risks, such as schedule slippage and cost overrun. After testing the build, at the end of first iteration, the customer evaluates the software and provides feedback.

The following illustration is a representation of the Spiral Model, listing the activities in each phase.



Spiral Model Strengths

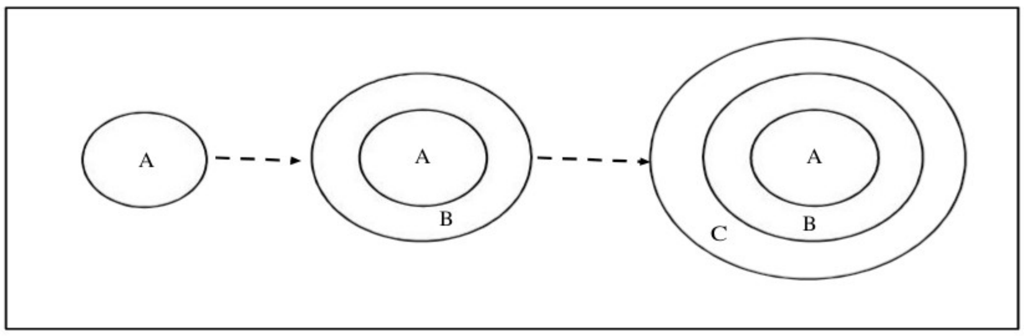
- ✓ Provides early indication of insurmountable risks, without much cost
- ✓ Users see the system early because of rapid prototyping tools
- ✓ Critical high-risk functions are developed first
- ✓ The design does not have to be perfect
- ✓ Users can be closely tied to all lifecycle steps
- ✓ Early and frequent feedback from users
- ✓ Cumulative costs assessed frequently

Spiral Model Weaknesses

• Time spent for evaluating risks too large for small or low-risk projects
• Time spent planning, resetting objectives, doing risk analysis may be excessive
• The model is complex
• Risk assessment expertise is required
• Spiral may continue indefinitely
• Developers must be reassigned during non-development phase activities
• May be hard to define objective, verifiable milestones that indicate readiness to proceed through the next iteration

When to use Spiral Model:
• When creation of a prototype is appropriate
• When costs and risk evaluation is important
• For medium to high-risk projects
• Long-term project commitment unwise because of potential changes
• Users are unsure of their needs
• Requirements are complex
• New product line
• Significant changes are expected (research and exploration)

The Evolutionary Model
This model is also known as the successive versions model. In this model the system is first broken down into several modules or functional units that can be incrementally implemented or delivered. In this model, each successive version of the product is a functioning system capable of performing some more useful work.



Evolutionary development of a Software Product

A, B and C are modules of a Software Product

Advantages
- ✓ The user gets a chance to experiment with a partially developed system much before the fully developed version is released
- ✓ The evolutionary model facilitates to elicit the exact requirements of the user for incorporating into the fully developed system

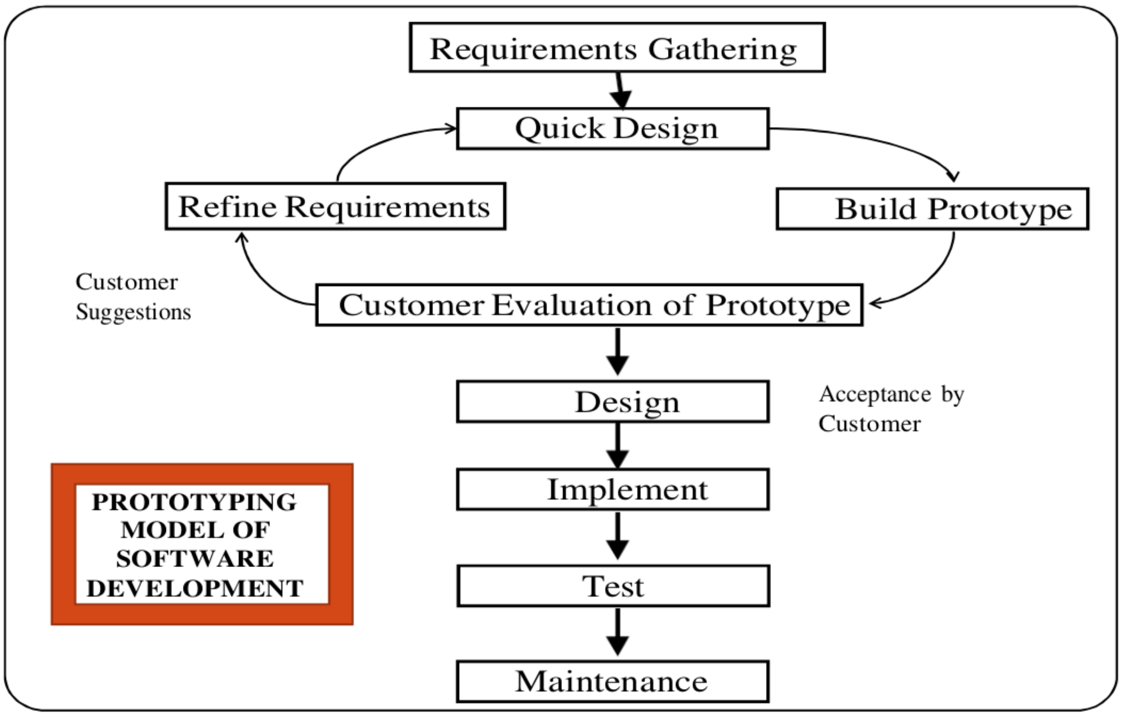
- ✓ Also, the core modules get tested thoroughly, thereby reducing chances of errors in the final product.

Disadvantages

- ✓ For most practical problems, it is difficult to subdivide the problem into several functional units that can be incrementally implemented and delivered.
- ✓ Evolutionary model is useful only for very large problems were it is easier to identify modules for incremental implementation.

The Rapid Prototyping Model

Rapid prototyping emphasises requirements analysis and validation, also called:
”customer oriented development”



Advantages

1. Reduces risk of incorrect user requirements
2. Good where requirements are changing/uncommitted
3. Regular visible progress aids management
4. Supports early product marketing

Disadvantages

1. An unstable/badly implemented prototype often becomes the final product.
2. Requires extensive customer collaboration
3. Difficult to know how long project will last.
4. Easy to fall back into code-and-fix without proper requirements analysis, design, customer evaluation and feedback.

RAD Model:

RAD model is Rapid Application Development model. It is a type of incremental model. In RAD model the components or functions are developed in parallel as if they were mini projects. The development are time boxed and then assembled into a working prototype.

Different phases in RAD model

- ✓ Business modeling
- ✓ Data modeling
- ✓ Process modeling
- ✓ Application generation
- ✓ Testing and turnover

End User Development Model

End user development refers to the activity of building the information system in terms of the user. Some types of information systems can be developed is called end-user with little or no formal assistances from technical specialists. This phenomenon is called end-user development. A series of software tools categorized as fourth generation languages makes this possible.

Categories of fourth generation language

- ✓ PC software tools
- ✓ Query language
- ✓ Report generator
- ✓ Graphics language
- ✓ Applications generator
- ✓ Application software package
- ✓ Very high level programming language

Outsourcing Model

Outsourcing is subcontracting a process, such as product design or manufacturing, to a third-party company. If a firm does not want to use its internal resources to build or operate information systems, it can outsource the work to an external organization that specializes in providing these services. Application service providers (ASPs) are one form of outsourcing.

Functional Information systems

A functional information system is a system that provides detailed information for a specific type of operations activity or related group of activities, as well as summarized information for management control of such activities.

The major functional systems of many organizations are

- ✓ Marketing information system
- ✓ Manufacturing information system
- ✓ HR information system
- ✓ Accounting information system
- ✓ Financial information system

Marketing Information system

MKIS is a computer based system that works in conjunction with other functional information systems to support the firm's management in solving problems that relate to marketing of the firms products.

Manufacturing information system

Manufacturing information system is a complete set of tools for managing the flow of manufacturing production data throughout the enterprise. This IS was designed to provide tools for both IT and operations personnel who would deliver services to anyone in the plant.

Human Resource Information System
HRIS refers to the systems and processes at the intersection between human resource management and information technology. It provides a method, by which an organization collects, maintains analyses and reports information on people and jobs.

Accounting information systems
AIS is the system of records a business keeps to maintain its accounting system. This includes the purchase, sales, and other financial processes of the business. The purpose of AIS is to accumulate data and provide decision makers (investors, creditors, and managers) with information to make decisions.

Financial Information System
The term FIS is used to describe the CBIS subsystem that provides information to persons and groups both inside and outside the firm concerning the firm's financial matters. Information is provided in the form of periodic reports, special reports, and results of mathematical simulation, electronic communications, and the advice of expert systems.

Decision Support System
DSS refers to a class of systems, which support the process of making decisions. The emphasis is on support rather than on automation of decision. Decision support systems allow the decision maker to retrieve data and test alternative solutions during the process of problem solving. DS is a specialized MIS designed to support an executives, skills at all stages of decision making i.e., problem identification, selecting relevant data, picking the approach to be used in decision making and evaluating the alternative courses of action.

Characteristics of DSS

- ✓ Provide rapid access to information
- ✓ Handle large amount of data from different sources
- ✓ Provide report and presentation flexibility
- ✓ Offer both textual and graphical orientation and Support drill-down analysis
- ✓ Perform complex, sophisticated analysis and comparisons using advanced software packages.

Activities of DSS

- ✓ What-if analysis
- ✓ Goal oriented
- ✓ Risk analysis
- ✓ Model building
- ✓ Graphical analysis

Components of DSS

- ✓ Data management sub system
- ✓ Model management sub system
- ✓ User interface sub system
- ✓ Knowledge based management sub system

Classification of DSS

- ✓ File drawer systems
- ✓ Data analysis systems
- ✓ Accounting models
- ✓ Representational models
- ✓ Optimization models
- ✓ Suggestion models

Advantages of DSS

- ✓ Improving personal efficiency
- ✓ Improving problem solving
- ✓ Facilitating communications
- ✓ Promoting learning or training
- ✓ Increasing organizational control

Disadvantages of DSS

- ✓ Limited storage capability
- ✓ Slow
- ✓ Limited information sharing
- ✓ Difficult
- ✓ Require extensive knowledge
- ✓ Translation problem
- ✓ Confliction

Executive Information Systems

Executive information systems are information systems that combine many of the features of management information systems and decision support systems. When they were first developed, their focus was on meeting the strategic information needs of top management. Thus, the first goal of executive information systems was to provide top executives with immediate and easy access to information about a firm's critical success factors (CSF), that is, key factors that are critical to accomplishing an organization's strategic objectives.

Features of EIS

- ✓ Drill-down capabilities
- ✓ Designed with managements CSF in mind
- ✓ Status access, trend analysis and exception reporting
- ✓ Personalized analysis
- ✓ Navigation of information
- ✓ Presents graphical tabular and textual information

Components of EIS

- ✓ Hardware
- ✓ Software
- ✓ Interface
- ✓ Telecommunication

Advantages of EIS

- ✓ Easy for upper level executives to use
- ✓ Ability to analyze trends
- ✓ Contribution to strategic control
- ✓ Ease access to existing information

- ✓ Instruments of change
- ✓ Better reporting system
- ✓ Improved mental model of the business for executives

Disadvantages of EIS

- ✓ Functions are limited, cannot perform complex calculations
- ✓ Hard to quantify benefits and to justify implementation of an EIS
- ✓ Difficult to keep current data
- ✓ May lead to less reliable and insecure data
- ✓ Small companies may encounter excessive costs for implementation

Knowledge Management System

Knowledge management is the process of identifying, collecting, preserving and transforming information into knowledge that is readily accessible in order to foster innovation and improve the performance of the organization. It is based on the assumption that the potential for sustained improvement exists in the knowledge derived frompeople, processes, designs and ideas within the organization. Knowledge management also implies the creation of a culture and structure that promotes information sharing and learning.

Features of KMS

- ✓ Purpose
- ✓ Context
- ✓ Processes
- ✓ Participants
- ✓ Instruments

Components of KMS

- ✓ Business process management
- ✓ Content management
- ✓ Web content management
- ✓ Knowledge applications management

Types of KMS

- ✓ Enterprise wide system
- ✓ Knowledge work system
- ✓ Intelligent techniques

Advantages of KMS

- ✓ Information sharing
- ✓ Elimination of redundancy of work
- ✓ Self learning

Disadvantages of KMS

- ✓ Justification of investment in knowledge management
- ✓ Obtaining senior management support
- ✓ Overcoming cultural hurdles to sharing
- ✓ Encouraging employees to use and share knowledge
- ✓ Confidentiality issues may limit ability to share knowledge
- ✓ Not a static system

Geographic Information Systems (GIS)

The work geographic in GIS carries two meanings
i. Earth: it implies that all data in the system are pertinent to earth's features and resources, including human activities based on or associated with these features and resources.
ii. Geographic space: it means that the commonality of both the data and the problems that the system are developed to solve is geography, i.e., location, distribution, pattern and relationship within a specific geographical reference framework.
GIS integrates hardware, software, and data for capturing, managing analyzing, and displaying all forms of geographically referenced informations.

Purposes of GIS
- ✓ To support decision making based on spatial data
- ✓ To support general research
- ✓ To collect, manipulate, and use spatial data in database management
- ✓ To produce standardized and customized cartographic production.

Capabilities of GIS
- ✓ Organization
- ✓ Queries
- ✓ Prediction
- ✓ Analysis
- ✓ Visualization
- ✓ Combination
- ✓ Basic notation

International Information Systems (IIS)
IIS are a general class of computer networks that operate in more than one nation-state. General IIS can be distinguished from more specific systems through their linkage to functionality, the sole limiting criteria being the providing of informational support to transactions that originate in one nation-state and terminate in another. The architecture of international systems is not fixed, but rather may be either centralized or decentralized on an international basis. In this type of information system, the important element is the existence of data crossing international borders in support of a transaction, typically trade and commerce data.
However, the nature of the trade, once it is made, limits it within the national border of at least one host nation-state.

Major dimensions for IIS
- ✓ Global environment
- ✓ Corporate global strategies
- ✓ Structure of the organization
- ✓ Management and business process and
- ✓ Technology platform

Types of IIS
- ✓ Transnational Information systems
- ✓ Global information systems
- ✓ Collaborative or cooperative information systems

Components of IIS
- ✓ Data input (Internal and External environment input

- ✓ Operational components (system controls, DBMS, user interface systems)
- ✓ Data output (sales management, senior management output)

Global strategies and business organization

- ✓ Domestic exporter strategy
- ✓ Multinational strategy
- ✓ Franchisers
- ✓ Transnational strategy

Challenges of IIS

- ✓ Technological challenges
- ✓ Regulations and tariffs
- ✓ Differences in payment mechanisms
- ✓ Language differences
- ✓ Cultural differences
- ✓ Conflicting economic, and security interests
- ✓ Political challenges
- ✓ Different standards

TOPIC 2: SYSTEM ANALYSIS AND DESIGN
Case tools – System Flow Chart, Decision Table, Data Flow Diagram (DFD), Entity Relationship (ER), Object Oriented Analysis and Design (OOAD), UML Diagram

SYSTEM
A system is an orderly grouping of interdependent components linked together according to a plan to achieve a specific objective.
System Analysis: It is the process of gathering and interpreting facts, diagnosing problems, and using the information to recommend improvements to the system.
Analysis specifies "what" the system should do, that is it sets the objective and Design states "how" to accomplish this objective.
Reasons for initiating System Analysis
- Problem solving
- New requirement
- Implement a new idea or technology
- Broad Systems improvements

System Design: It is the process of planning a new business system or complement an existing system.
System design work begins after systems analysis is completed. System analysis lists users' requirements. Now it is the task of systems designer to identify data requirements and data sources. The system design may be divided into conceptual / logical design and physical design.
Logical design: The logical design of a system pertains to an abstract representation of the data flows, inputs and outputs of the system.
Physical design: The physical design relates to the actual input and output processes of the system.

CASE TOOLS (Computer Aided Software Engineering Tools)
With the increasing speed of changing market demands new products replace old ones much earlier than before, so the development of new products has to go faster. So, to speed up the software system building process, an introduced term in field of software engineering is CASE tools. It is a generic term used to denote automated support associated with the software development activities. CASE supports the development, verification, maintenance and evolution of processes and artifacts.
CASE tools are software programs that are designed to assist human programmers with the complexity of the processes and the artifacts of software engineering. These automated tools help in synthesis, analysis, modeling, documentation, coding etc.

Characteristics of CASE tools
- ✓ Standard methodology
- ✓ Flexibility
- ✓ Strong integration
- ✓ Integration with testing software
- ✓ Support for reverse engineering
- ✓ On-line help

Classes of CASE tools:
- ✓ Analysis, design and specification tools
- ✓ Data modeling tools
- ✓ Prototyping tools

- ✓ Coding tools
- ✓ Testing tools
- ✓ Implementation tools
- ✓ Upper CASE tools
- ✓ Lower CASE tools

Architecture of CASE environment

- ✓ A database (to store the information)
- ✓ An object Management system (to manage changes to the information)
- ✓ A tools control mechanism (to coordinate the use of CASE tools)
- ✓ A user interface (to provide consistent pathway between actions made by user and tools)

Advantages of CASE tools

- ✓ Easy revision of system descriptions and graphic representations
- ✓ Support of system prototyping through the capability to change specifications
- ✓ Some CASE tools have the capability to produce working source code for the application
- ✓ A CASE tool also provides maintenance support as a result of storage of
- ✓ system specification in a central information repository.

Disadvantages of CASE tools

- ✓ Absence of standard levels of methodology support
- ✓ Limited functions supported
- ✓ Conflicting use of diagrams
- ✓ Human tasks remain critical

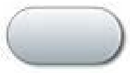
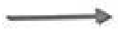
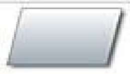
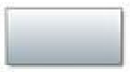
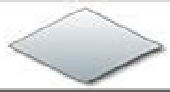
SYSTEM FLOW CHART

A Flow chart is a pictorial representation of the sequence of operations in systems.

Flowcharting is the most common method of describing procedures in a computer based information system.

A flow chart is the plan to be followed when a program is written. It acts like a road map for a programmer and guides him/her in proceeding from the starting point to the final point in a logical manner.

Flow Chart Symbols:

Symbol	Name	Function
	Start/end	An oval represents a start or end point.
	Arrows	A line is a connector that shows relationships between the representative shapes.
	Input/Output	A parallelogram represents input or ouptut.
	Process	A rectangle represents a process.
	Decision	A diamond indicates a decision.

Rules for drawing flow chart

- ✓ First formulate the main line of logic, then incorporate the details
- ✓ Maintain a consistent level of detail for a given flowchart.
- ✓ Do not give every detail on the flowchart
- ✓ Words in the flowchart symbols should be common statements for easy understanding
- ✓ Be consistent in the use of names and variables in a flowchart
- ✓ Go from left to right and top to bottom while constructing a flowchart
- ✓ Keep the flowchart as simple as possible
- ✓ The crossing of flow lines should be avoided
- ✓ If a new page is needed for drawing a flowchart, it is recommended that the flowchart be broken at an input or out point
- ✓ Finally check whether the flowchart is logically correct and complete.

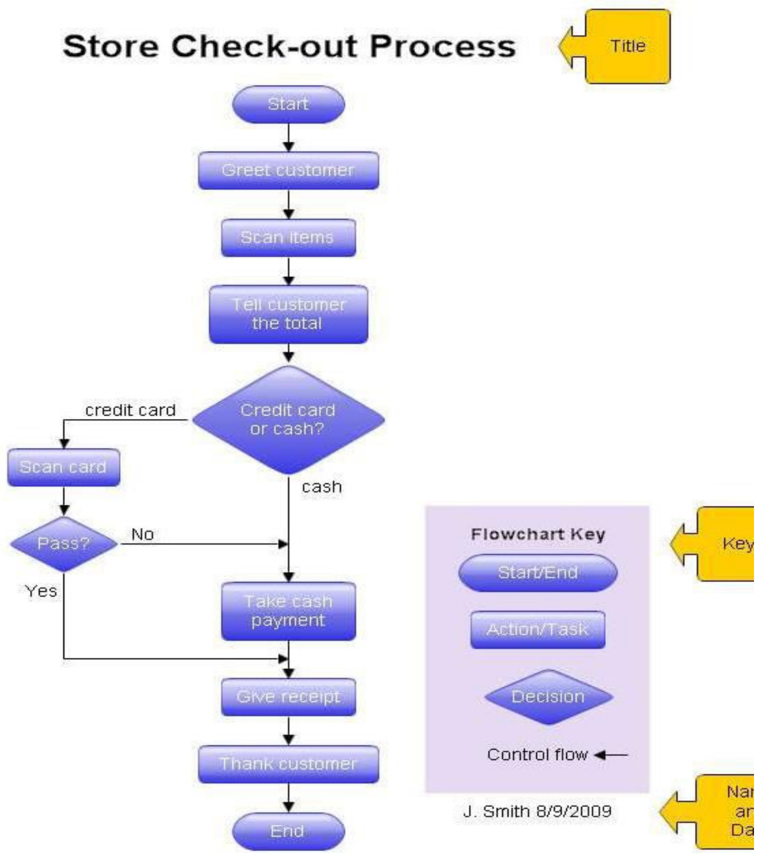
Advantages of flow charts

- ✓ Conveys better meaning
- ✓ Effective joining of a part of a system
- ✓ Efficient coding

Disadvantages of flow charts

- ✓ Takes more time to **draw**
- ✓ Difficult to make changes
- ✓ Non-standardization

Examples of Flow Charts:



DECISION TABLE

Decision tables are a precise yet compact way to model complicated logic. Decision table is a matrix representation of the logic of a decision, which specifies the possible conditions for the decision possible conditions for the decision and the resulting
actions.

Decision tables are a convenient way to organize information in a systematic manner.

A major drawback of a decision tree is the lack of information in its format to tell what other combinations of conditions to test. This is where the decision table is useful.

Creating Decision tables

- ✓ Name conditions and values that each condition can assume
- ✓ Name all possible actions that can occur
- ✓ List all possible rules
- ✓ Define actions for each rule
- ✓ Simplify decision table

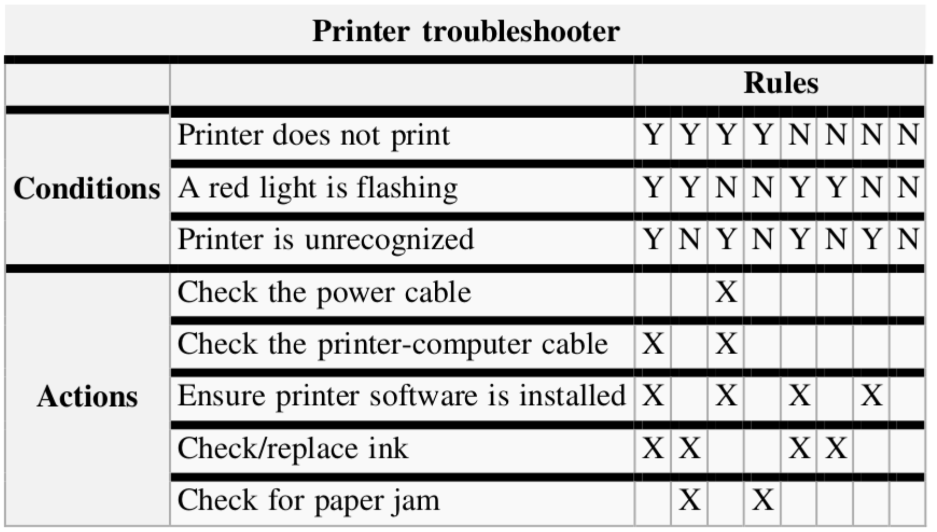
Advantages of decision tables

- ✓ Decision rules are clearly structured
- ✓ Managers can be relieved from 5 decision making
- ✓ Consistency in decision making☔ Communication is easier between managers and analysts
- ✓ Documentation is easily prepared, changed, or updated,
- ✓ Easy to use
- ✓ Easier to draw or modify compared to flowcharts
- ✓ Facilitate more compact documentation

Disadvantages of decision tables

- ✓ Impose an additional burden
- ✓ Do not depict the flow
- ✓ Not easy to translate and
- ✓ Cannot list all the alternatives

The following are the balanced decision table.



Printer troubleshooter

		Rules							
Conditions	Printer does not print	Y	Y	Y	Y	N	N	N	N
	A red light is flashing	Y	Y	N	N	Y	Y	N	N
	Printer is unrecognized	Y	N	Y	N	Y	N	Y	N
Actions	Check the power cable			X					
	Check the printer-computer cable	X		X					
	Ensure printer software is installed	X		X		X		X	
	Check/replace ink	X	X			X	X		
	Check for paper jam		X		X				

DATA FLOW DIAGRAM

DFD are widely used graphic tools for describing the movement of data within or outside the system. These diagrams, popularly called as DFDs, quickly convey to both the software developers and users, how the current system is working and how the proposed system will work.

The main advantage of DFD is that they are easily understood by the users and hence, users can suggest modification in the proposed system about whether processes will operate in sequence or in parallel. It is therefore quite different from a flow chart.

Meaning of **Flow chart:** A diagram consisting of a set of symbols such as rectangles or diamonds and connecting lines that shows step-by-step progression through a procedure, processor system

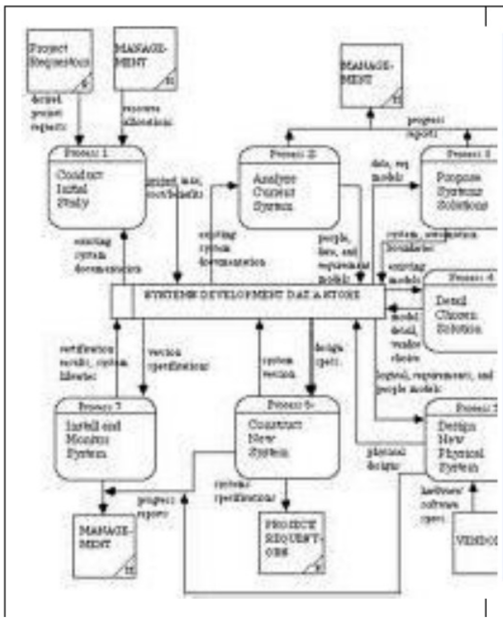
Components of DFD:

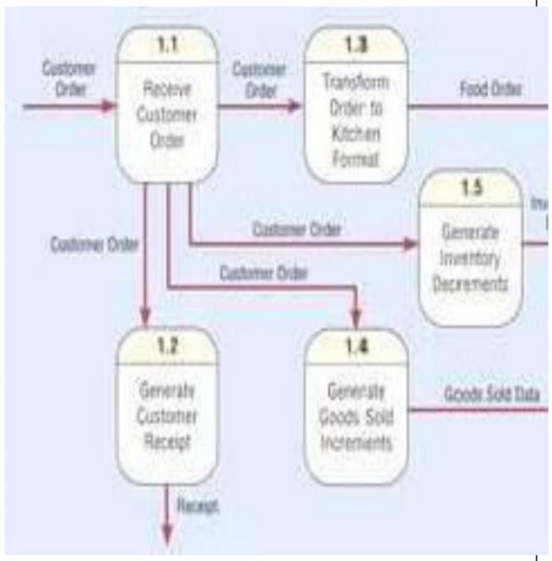
➢ External Entity	□ External Entity
➢ Process	▢ Process
➢ Data Source	⊏ Data Store
➢ Data Flow	→ Data/Information Flow

Rules for drawing a DFD:

- ✓ Sources cannot leak data directly to a data store
- ✓ A data store cannot pass the data directly to a destination
- ✓ A data flow out of a data store is read only.
- ✓ Data cannot flow directly from one data store to another
- ✓ Any process producing output by itself should be a source.
- ✓ Any process with only input should be a data destination.
- ✓ Each subsystem must be a process on the next higher level diagram.

Examples of DFDs





Advantages of DFD

- ✓ Early implementation
- ✓ Study independence
- ✓ Analysis
- ✓ Tool for communication
- ✓ Reduces costs

Disadvantages of DFD

- ✓ Imprecise
- ✓ Absence of control aspects
- ✓ Highly subjective

ERD - Entity Relationship Diagram

An entity-relationship (ER) diagram is a specialized graphic that illustrates the interrelationships between entities in a database. An E-R diagram expresses the overall logical structure of a database graphically.
The entity-relationship diagram (also known as an ERD) is a network model that describes the stored data layout of a system at a high level of abstraction.
It is quite different from the dataflow diagram, which models the functions performed by a system, and it is different from the state-transition diagram,, which models the time dependent behavior of a system

Objectives of ERD

- ✓ Straight forward relational representation
- ✓ Easy conversion of ER to other 8 data model
- ✓ Graphical representation for better understanding

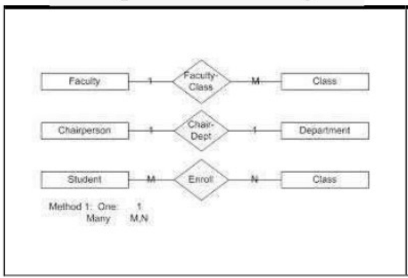
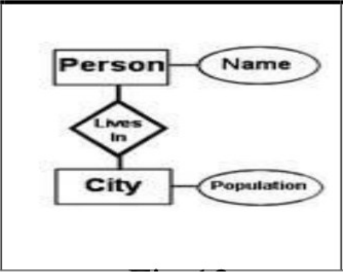
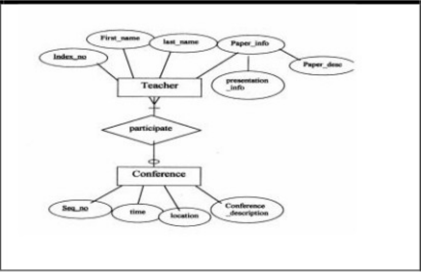
The components of an ERD:

- ✓ Entity / Object
- ✓ Attributes
- ✓ Relationships

One to One
One to Many
Many to Many

- ✓ Associative object type indicators
- ✓ Super type / subtype indicators

Examples of ER Diagram

 Faculty Faculty-Class Class 1 M Chairperson Chair-Dept Department Student Enroll N Method 1: One 1 Many M,N		 First_name last_name Paper_info Index_no Paper_desc Teacher presentation_info participate Conference Seq_no time location Conference_description

OBJECT ORIENTED ANALYSIS AND DESIGN (OOAD)

Object oriented design methodology focuses attention not on the function performed by the program but instead on the data that are to be manipulated by the program.

Object oriented technology is one of the latest approaches to software development and its shows much promise in solving the problems associated with building modern software systems.
In the object oriented design approach, the system is viewed as a collection of objects. The system states decentralized among the objects and each object manages its own state information.
Object oriented concepts (Six basic concepts)

- ✓ Class (it is a template for building objects. It is used as a blue print to create objects and it includes attributes and methods that are created objects all share.
- ✓ Object (a system is designed as a set of interacting objects and each represents a tangible real world entity)
- ✓ Information Hiding (Ability to protect some components of the object from external entities. This is realized by language keywords to enable a variable to be declared as a private to the owning class.)
- ✓ Inheritance (Ability for a class to extend or override functionality of another class)
- ✓ Interface (Ability to defer the implementation of a method)
- ✓ Polymorphism (Ability to replace an object with its sub objects)

Data and process modeling
Object oriented data modeling is ba 9 sed on what is called the "object oriented paradigm", which is not just a way of programming but most importantly, is a way of thinking abstractly about a problem using real-world concepts, rather than implementation oriented concepts.
Object oriented data models achieve these requirements by providing appropriate mechanisms to represent the structure of application domains with a high degree of accuracy while also placing emphasis on operational abstractions
To represent the data and process modeling, the most common used language is UMS. UML stands for Unified Modeling Language.

UNIFIED MODELING LANGUAGE (UML)
Unified Modeling Language (UML) combines techniques from data modeling (entity relationship diagrams), business modeling (work flows),object modeling, and component modeling. It can be used with all processes, throughout the software development life cycle, and across different implementation technologies.
The Unified Modeling Language (UML) offers a standard way to visualize a system's architectural blueprints, including elements such as:

- ✓ activities
- ✓ actors
- ✓ business processes
- ✓ database schemas
- ✓ (logical) components
- ✓ programming language statements
- ✓ Reusable software components.

UML has synthesized the notations of the Booch method, the Object-modeling technique (OMT) and Object-oriented software engineering (OOSE) by fusing them into a single, common and widely usable modeling language. UML aims to be a standard modeling language which can model concurrent and distributed systems.
UML models may be automatically transformed to other representations (e.g. Java) by means of QVT-like transformation languages. UML is extensible, with two mechanisms for customization: profiles and stereotypes.

Software development methods
UML is not a development method by itself; however, it was designed to be compatible with the leading object-oriented software development methods of its time (for example OMT, Booch method, Objector). Since UML has evolved, some of these methods have been recast to take advantage of the new notations (for example OMT), and new methods have been created based on UML, such as IBM Rational Unified Process (RUP). Others include Abstraction Method and Dynamic Systems Development Method.

UML Diagram
A Diagram is the graphical presentation of a set of elements, most often rendered as a connected graph of verticers (things) and paths (relationship).
UML diagrams represent two different views of a system model:
- **Static (or structural) view:** emphasizes the static structure of the system using objects, attributes, operations and relationships. The structural view includes class diagrams and composite structure diagrams.
- **Dynamic (or behavioral) view**: emphasizes the dynamic behavior of the system by showing collaborations among objects and changes to the internal states of objects. This view includes sequence diagrams, activity diagrams and state machine diagrams.

UML models can be exchanged among UML tools by using the XML Metadata Interchange (XMI) interchange format.

STRUCTURE DIAGRAMS
Structure diagrams emphasize the things that must be present in the system being modeled. Since structure diagrams represent the structure, they are used extensively in documenting the software architecture of software systems.
- **Class diagram**: describes the structure of a system by showing the system's classes, their attributes, and the relationships among the classes.
- **Component diagram**: describes how a software system is split up into components and shows the dependencies among these components.
- **Composite structure diagram:** describes the internal structure of a class and the collaborations that this structure makes possible.
- **Deployment diagram**: describes the hardware used in system implementations and the execution environments and artifacts deployed on the hardware.
- **Object diagram**: shows a complete or partial view of the structure of an example modeled system at a specific time.
- **Package diagram:** describes how a system is split up into logical groupings by showing the dependencies among these groupings.
- **Profile diagram:** operates at the meta model level to show stereotypes as classes with the <<stereotype>> stereotype, and profiles as packages with the <<profile>> stereotype. The extension relation (solid line with closed, filled arrowhead) indicates what meta model element a given stereotype is extending.

BEHAVIOR DIAGRAMS
Behavior diagrams emphasize what must happen in the system being modeled.
Since behavior diagrams illustrate the behavior of a system, they are used extensively to describe the functionality of software systems.
- **Activity diagram**: describes the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.
- **UML state machine diagram**: describes the states and state transitions of the system.

☔ **Use Case Diagram:** describes the functionality provided by a system in terms of actors, their goals represented as use cases, and any dependencies among those use cases.

INTERACTION DIAGRAMS

Interaction diagrams, a subset of behavior diagrams, emphasize the flow of control and data among the things in the system being modeled:

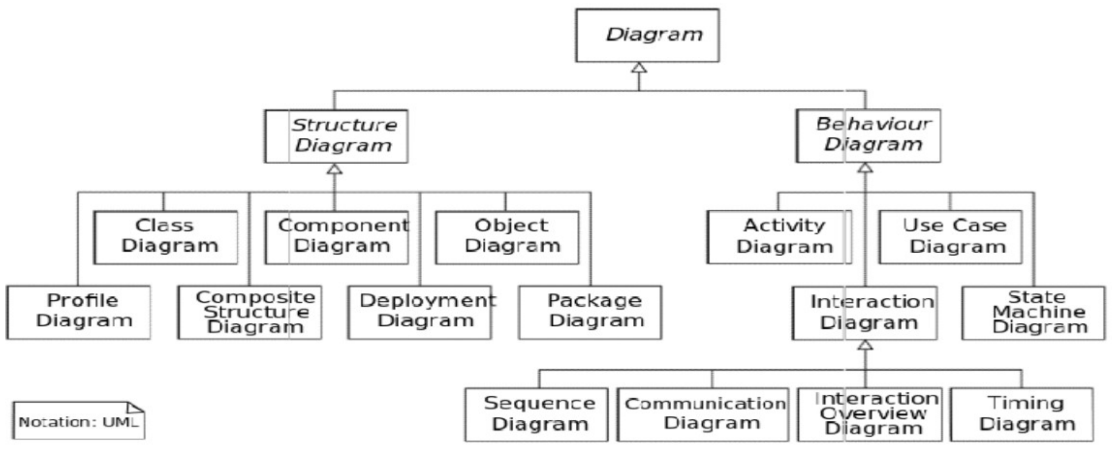
☔ **Communication diagram**: shows the interactions between objects or parts in terms of sequenced messages. They represent a combination of information taken from Class, Sequence, and Use Case Diagrams describing both the static structure and dynamic behavior of a system.

☔ I**nteraction overview diagram**: provides an overview in which the nodes represent communication diagrams.

☔ **Sequence diagram**: shows how objects communicate with each other in terms of a sequence of messages. Also indicates the life spans of objects relative to those messages.

☔ **Timing diagrams**: a specific type of interaction diagram where the focus is on timing constraint s .

UML Diagrams Overview:



TOPIC 3– DATABASE MANAGEMENT SYSTEMS

DBMS - HDBMS, NDBMS, RDBMS, OODBMS, Query Processing, SQL, Concurrency Management, Data warehousing and Data Mart.