UNIT-1 (Foundations for Systems Development)

System → collection of components to realize some objective. E.g. library system.
Activities: input, processing, output.
Additional activities: feedback and control.

Information System (IS) → system to provides info to people in an organization.
IS capture and manage data to support org and employees.
Activities: input → collects raw data
processing → converts raw data into meaningful info.
output → transfers this info to people.

R

Types of IS → TPSs, MISs, DSSs, EISs, ES, C&Cs

फिल्म मिस डिस इस इस सिस

- TPSs → computerized system that records daily routine transactions e.g. hotel reservation system, employee record keeping etc.
- MISs → management level systems and serve mangement level functions like planning, controlling and decision-making. e.g. Salary analysis
- DSSs → These also serve at management level. Uses internal info from TPSs and MISs. Has more power. e.g. Contract cost analysis
- EISs → strategic level, manage data about external events such as new tax laws, e.g. 5-year operating plan.
- ES → extension of DSS, deals with logic and reasoning. Implemented with AI technology
- C&Cs → Enable effective communication between workers, customers. Use network technology for effici

System Analysis and Design:
team to develop and maintain computer based IS, Important result of SAD is application software. Designed to support organizational functions and process.
We use methodologies, Techniques & Tools.
methodologies → ~~Multi step~~ Multi-step comprehensive approaches
Techniques → ensures work is well through-out
Tools → assist in application of techniques.

Modern Approaches to SAD: SAD methodologies started during 1950 to 1960. Reasearchers argued crisis was due to lack of discipline of programmers. New technologies and approaches were developed after 1970-1990. OOP, case tools etc used. ~~He~~ Web App Dev, Wireless PDA's, component based applications are modern approach to SAD

SDLC → ~~common~~ methodology for system development. P, A, D, Imp, Main

LZITS important main

- P → identifying needs
- A → requirement determination & analysis study. (structural requirements)
- D → recommended plan into logical and physical syste [what can be improved or replaced from project]
- Imp → coded, tested, installed.
- Main → fix issues and impro

Heart of System Development Process: It is analysis-design-implementation of SDLC i.e, A, D, Imp
~~Analysis~~ →.

Traditional Waterfall SDLC:
Planning - Analysis - Logical Design, Physical design, Implementation, Maintenance

→ One phase begins when another completes, little backtracking and looping.
→ Logical and physical design, Linear sequential approach to SDLC
→ Used if requirements not changing frequently, Application is not complicated and big. Project short, requirements clear, environment stable, resources available and trained.

✓ CASE tools → automated tools used by systems analysts to develop IS. For increasing productivity and improving quality of systems. Software packages to automate activities in SDLC. Range from simple diagramming tools to sophisticated programs.

Types → Diagram tools, Analysis Tools, ~~Computer display and Report generators~~, Documentation generators, Code Generators. Central Repository.

Components → Upper-case, Lower-case, Integrated tools, Central Repository.

- Upper-case ↓ P, A, D of SDLC
- Lower-case ↓ Imp, Main of SDLC.
- Integrated tools ↓ All stages of SDLC from gathering requirements to testing and documentation.
- Central Repository ↳ Single point storage for diagrams, reports and documents.

→ Phases Ⓡ

✓ Approaches to SAD: Prototyping, Spiral Approach, RAD Approach, Agile development Approach.

Prototyping ↔ only produce what users want. Do not develop complete system at once. Create prototype, present to user and refine it according to requirements.
Phases: Requirement gathering, Quick Design, Build Prototype, User Evaluation, ~~Engineer Product~~ Refining prototype, Engineer Product.

Spiral Approach ↕ Looks like spiral with many loops, exact no. of loops in spiral are unknown, and can vary from project to project. Each loop of spiral ~~is~~ called phase of Software dev process. Each phase divided into four quadrants.
Phases: Obj. determination and identifying alternate solutions, Identify and resolve risks, develop next version of product, review and plan for next phase.

RAD Approach | O-O approach includes a method of development as well as ~~of~~ software tools. meets rapidly changing business requirements closely. Some devs. see it is helpful approach for ecommerce.
Phases
→ Requirements Planning
→ RAD Design Workshop → design and refine phase
→ Implementation

Agile development Approach ↓ Intended for devs builds project which can adopt transforming requests quickly. Developed to make rapid and easy project. Activities and Behaviours ----
Phases:
→ Exploration (estimating time & other)
→ Planning (agree on date).
→ Iteration to first release (skeleton or outline of project)
→ Productionizing (features added and product release)
→ Maintenance

Project Management and Its Phases: Controlled process of phases (Initiating, Planning ....). Important aspect of development of IS. Ensures that project meets customer expections and are delivered within budget and time constraints.

Phases: Initiating, Planning, Executing, Monitoring and Control, Closing.

- Initiating ↑ defining project goal, scope, potential risks etc.
- Planning ↑ creating blueprint for time and scope
- Executing ↕ Developing project by project manager with team members in time.
- Monitoring and Control ↑ Project manager records progress, Monitors – corrective action in project to keep project in track.
- Closing ↳ Project manager closes contracts. Documentation will be archieved and final report will be produced.

Managing IS Project: Shaping a Project, Project Triangle, Project Manager.

- Shaping a Project ↑ shape project on the basis of time, budget, quality and deliver with meeting user requirements.
- Project Triangle ↑ challange is to find optimal balance among three factors cost, scope and time. any factor can affect other.
- Project Manager ↳ System anylist with diverse set of skills → leadership, Management, Techical problem solving, Conflict Management, Customer Relations. Project Manager is responsible for Project Management (Initiating, Planning, ...).

Gantt Charts vs. Network Diagrams: Defn, task, depicts, slack time, category.

Calculating Expected time durations using PERT: technique that uses optimistic, pessimistic and realistic time estimates to calculate Expected time. Helps to obtain better time estimate when we are uncertain how much time it takes to complete. $\boxed{ET = \frac{o + 4r + p}{6}}$

Project Management Software: variety of softwares used. New tools are released by software vendors. Trello, Asana, Jira, ClickUp etc are some project management tools.

⊛ Why do we need SAD, IS?
⊛ What are forward, reverse and round-trip engineering?

Unit-3

Analysis

Done by System Analyst

Determining system requirements

(includes collection of information and understanding user's requirements).

Traditional Methods
- Interviewing & Listening
- Group Interviewing
- Directly observing users.
- Analyzing Procedures and other documents.

Contemporary Methods
- Joint Application Design (JAD)
- Prototyping

Radical Methods
- Business Process Reengineering (BPR)
- Identification of process to reengineer
- Disruptive Technologies

Structuring system requirements.

Data Flow Diagrams (DFDs)

(A common form of process model that involves graphicall representing the process or actions, that capture, store and distribute data betwn system and it's environment)

Symbols/Components of DFD

1) Process
2) Data Flow
3) Data Store
4) External Entity.

Rules of Data Flow
- कुनै पनि entity or data store, processing बिना अरु entity or data store मा जादैन यदि गत हो,

Guidelines For developing DFD
- Context diagram Level-0.
- Unique Name for Levels
- No Cross Line in DFD
- Numbering Convention

⊛. Logical vs. Physical DFD's.

Logical:
- how it operates (conceptual)
- implementation ignored.
- Independent of particular technology.

Physical:
- implementation including hardware, software, files. logical model implemented to achieve goal of business.

Logic Modeling (graphic representation among relationships ....)
Components: Needs, Inputs, Activities, Outcomes

- ✓ Modeling logic with decision tables
  - based on conditions
- ✓ Modeling logic with decision trees.
  - flowchart like tree structure
  - node - test on an attribute.
  - branch - outcome of test
  - leaf - holds class label.
- Modeling logic with Pseudo-codes.
  - → rough draft of program
  - → no syntax
  - → bridge betw algo and program

ER-Modeling (graphical approach to ~~the~~ database design)
- defines data elements & their relationships.
- represents real-world objects

- Elements of ER-diagram (ER design issues)
  - ↳ Entity - thing or object with independent existence
    - ↳ Entity type ← collection of entities
    - ↳ Entity set. → same as entity type but defined at particular point of time.
  - ↳ Attributes ↔ particular properties that describe entity
    - Types
      - ↳ Atomic vs composite
      - ↳ Single vs multivalued
      - ↳ Key attribute.
- Symbols used in ER-diagram
- Relationships (Association among entities)
  - ↳ Unary relationship
  - ↳ Binary relationship.
  - ↳ N-ary relationship
- What is data modeling? How it is different from process modeling? How do you use entity relationship model for data modeling?

Unit-4
DESIGN

Database Design → Organization of data according to database model. Produces logical and physical designs.

Process of database design: normalization process is used to build a data model in logical database modeling and design with four key steps:

Relational Database Model: data in the form of tables (relation), [columns = attributes & rows = records (tuple)]. It is logical view of data. Eg: Student info table

Characteristics :- unique name of each relation, unique attributes, duplicate tuples not allowed, each tuple must have exactly one data value for an attribute, Tuples and attributes does not have to follow significant order.

Normalization: minimizing redundancy process, purification or filtering process for better design, Redundancy may cause insertion, updation & deletion anamolies.

Normal form → Degree to which relation is normalized.

Functional Dependency $X \rightarrow Y$
if $t1[X] = t2[X]$ then, there must be $t1[Y] = t2[Y]$.

1NF
↓
If it does not contain any multivalued attribute.

Examples of each NF 1st one is easy write 2NF and 3NF Questions in file

2NF
↓
if it is in 1NF and does not contain any P.D

key → unique identifier
candidate key → key whose proper subset is not key.
Prime attribute → attributes making candidate key
Non-Prime attribute → which are not making candidate key.
P.D = if proper subset of candidate key determines non-prime attribute.

3NF
↓
if it is in 2NF and there is no transitive dependency for non-prime attributes.

Transitive Dependency
If $A \rightarrow B$ and $B \rightarrow C$ are two FDs then $A \rightarrow C$ is called transitive dependency.

Transforming ER diagrams into Relations: Represent entities, Represent relationships, Normalize relations, Merge relations. [Re en, Re re., No re, Me re]
रिन, रिरि, नोरि, मेरि

Merging Relations: remove redundancy. Example of 3NF

Physical File & Database Design → for this info should be collected during SDLC phase.
It includes;
→ Normalized relations including volume estimates.
→ Defn of each attribute
→ Descriptions of when and where data are used.
→ Requirements for response and data integrity.
→ Descriptions of technologies used for implementing files and ~~data~~ designing database

Designing Fields:- - - - -

Designing Physical Tables: named set of rows and columns.
efficient use of secondry storage and data processing speed.

✓ Process of Designing Forms and Reports
User-focused activity typically follows prototyping approach.
→ First we collect initial requirements from users by several questions, who, what, when, where and how. use purpose
when needed · where delivered · how many people use or view
→ Then collected info is refined into initial prototype ... - - -

✓ Formatting Forms and Reports:-
1) General Formatting Guidelines: Meaningful titles, Meaningful information, Balanced layout, Easy Navigation.
2) Guidelines For Displaying Contents: Highlighting information, Using Color, Displaying text, Designing tables and Lists.

Paper Based vs. Electronic Reports: store, edit & copy, multiple user access, delivery, searching.

#Designing Interfaces and Dialogues:

focuses on how information is provided to and captured from users.
Process is similar to that of forms and reports i.e, user-focused activity and follows prototyping approach.

Measures of Usability: Learnability, Efficiency, Error rate, Memorability, Satisfaction.

1) Methods of Interacting: Command language interaction, Menu interaction, Form interaction, Object-based interaction, Natural Language interaction.

2) Designing Interfaces: (It is front-end application to interact with software)
It consists: Designing layouts, Structuring data entry, Controlling data input, Providing Feedback, Providing Help.

3) Designing Dialogues: process of designing the overall sequences that users follow to interact with an information system. Consists three major steps:
→ Designing dialogue sequence (how uses might interact with system).
→ Building a Prototype
→ Accessing Usability (consistent in form, function style).

4) Designing Interfaces and Dialogues in Graphical Environments:
→ Become expert of GUI environment.
→ Understand available resources and how they can be used.

UNIT-5

System Implementation

(way of carrying out a developed system into working condition).

Involves Activities: Coding, Testing, Installation, documenting system, & training and supporting users.

Coding → Physical design specifications into working computer code

Testing → Each module is tested, then part of larger program, then finally system.

Installation → Current System replaced by new system. Conversion of data & software to new system.

documenting system → Using guidelines, documenting each code, ~~ab~~ collecting, organizing, storing and maintaining complete record.

Training & Supporting Users →

Software Application Testing: checks functionality, verifying and validating bug free or not and technical requirements, efficiency, accuracy, & usability.

Types of Testing: Inspections, Desk checking, ~~Unit~~ Stub testing, Integration testing, System testing,

Inspections → programming language errors tested. Syntax, semantics and other error checked by automated inspection software

Desk checking → logic of program checked with paper and pencil

Stub testing → testing modules written and tested in top-down fashion.

Integration testing → Bringing together more than one module.

System testing → Bringing together all programs. Tested by black-box testing or white-box testing.

Differences betw^n black-box and white-box testing: internal structure of program (known/unknown) testers/developers, function testing/Structural testing, Programming knowledge (required/not required) Implementation knowledge (not required/required).

Installation types/approaches: Direct installation, Parallel installation, Single-location installation, Phased installation.

Documenting System → collecting, organizing, storing and maintaining complete record of each phase of development cycle.

System documentation → system's design specifications, internal workings & functionality. Intended for maintenance programmers, Internal & External documentation.

User documentation → written or visual information about system, how it works and how to use it. Contains operational details, security measures & problems that may arise.

Maintaining Information Systems → Correcting and Upgrading process of system to eliminate errors. Involves 4 major activities: Obtaining maintenance requests, Transforming requests into changes, Designing changes, Implementing changes.

Types of Maintenance: Corrective, Adaptive, Perfective, Preventive.

Corrective → repair defects

Adaptive → changes made to functionality with changing needs

Perfective → making enhancements & improve performance.

Preventive → reduce chance of failure in future

Cost of Maintenance: 60 to 80% of budget of company on maintenance, 52% programmers assigned in company for maintenance, only 3% assigned for new application development.

Numerous factors effect it like: no. of latent defects, no. of customers, documentation quality & tools used.

Managing Maintenance: Managing maintenance personnel & Measuring maintenance effectiveness

Factors influencing maintenance cost: Latent defects, no. of customers, system documentation quality, maintenance personnel, Tools, Well-structured programs.

UNIT-6

Introduction to Object-Oriented Development

Benefits/Characteristics of OOP system: Class, Abstraction, Inheritance, Polymorphism, Reusability, Data hiding.

- Class: collection of similar objects
- Abstraction: providing only needed info and hiding details
- Inheritance: Objects of one class acquire properties of another class
- Polymorphism: ability to make more than one form. for eg + operator
- Data hiding: Seperation of data from direct access by program

Object-Oriented System Analysis and Design (OOSAD)

Unified Modeling language: [UML]

General purpose language in software ~~[illegible]~~ engineering to provide standard way to visualize the system. Supports high level developments such as frameworks, patterns and collaborations. It includes collection of elements such as;
→ Programming language statements
→ Actors that specify a role played by user
→ Logical and Reusable software components etc.

UML divided into two categories: representing structural info (structural diagrams) & representing general types of behaviour (behavioural diagrams)

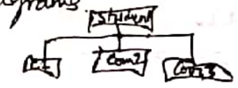
Structural Diagrams (represent static aspect of system)

Class Diagram

Consists: classes, interfaces, associations, & collaborations represent O-O view of system. Helps to identify relationship between classes & objects.
In UML class is separated by rectangle with 3 compartments: classname, list of attributes, & list of operations.

Object Diagram

Consists specific instances of classes and relationships. Similar to Class diagram except it shows instances of classes in system.
Derived from class diagrams so dependent on class diagrams.



Component Diagram

Consists classes, interfaces or collaborations.
Represents a set of components and their relationships.
Represents how physical components in system are organized. Components communicate with eachother using interfaces. Interfaces are linked by connectors.

Deployment Diagram

Consists a set of nodes and their relationships. These nodes are physical entities where the components are deployed. These diagrams show the implementation environment of the system used to represent system hardware and software.

Behavioural Diagrams: (represent dynamic aspect of system).

Use case diagram

Overview of actors involved in system. Represent case view of system. Used to show functionality of system.

Activity Diagram

Represent workflows in a graphical way. Used to describe business workflow of any component of system. Sometimes used as an alternative to state machine diagrams.

Sequence Diagram

Shows how objects interact with eachother and order. They show interactions for a particular scenario. The processes are represented vertically and interactions are shown as arrows. Used by businessman and Software developers.

State Diagram

Similar to activity diagrams, although notations and usage changes a bit. Very useful to describe behaviour of objects that act differently according to the state they are in at the moment.