System Design and Analysis

CS-3004

Introduction

Lecture # 01,02, 03
6,8, 9 Sep

Rubab Jaffar
rubab.jaffar@nu.edu.pk

Today’s Outline

- i4hnvpn
- Administrative Stuff
- Overview of 3004
- Introduction to system analysis and design
- Why study AD?

Administrative Stuff

About me

- Graduated from FJWU
- Complete Masters from NUST
- Research interests:
    - Software engineering
    - Database systems
    - Algorithm analysis

Administrative Stuff

Office hours

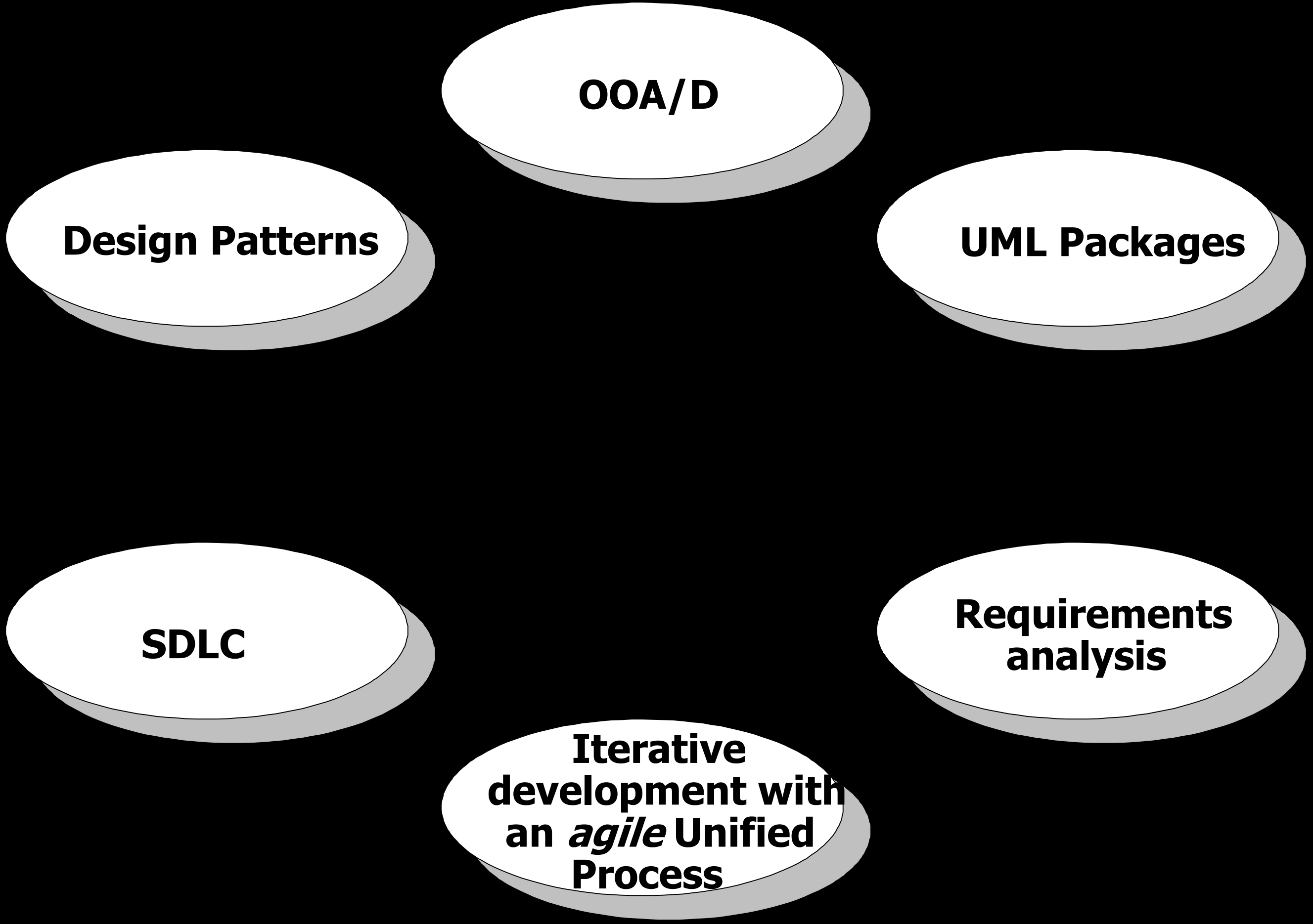
- Office 6 (CS building)
- Office hours: 2:00 to 3:00 pm
    - (Wednesday, Thursday, Friday)

About the course

- Study application of a software engineering approach that models and designs a system as a group of interacting objects.
- Various cases studies will be used throughout the course to demonstrate the concepts learnt.
- A strong in class participation from the students will be encouraged and required during the discussion on these case studies.

Overview of 3004

Major Topics of the course



Overview of 3004

Course Outline

- Course Introduction
- SDLC
- RUP
- Requirement Engineering
- UML Relationships
  - Association
  - Aggregation
  - Composition
- UML Packages
  - Use Case
  - Class Diagram
  - Activity Diagram
  - Collaboration Diagram
  - Sequence Diagram
  - State Transition Diagrams
- Design Patterns

Pre-requisites /Knowledge assumed

- Programming language concepts
- Data structure concepts
- We assume you have the skills to code in any programming language therefore you
  - can design, implement, test, debug, read, understand and document the programs.

Tentative Grading Policy

- Assignments/Class Participation: 10 %
- Mid Exam 1: 15 %
- Mid Exam 2: 15 %
- Final: 50%
- Presentations + Projects: 10%
- Absolute Grading Scheme

Project

- An advanced level project
  - Should provides interesting realistic problem,
- Small 3-4 sized team
- Emphasis on good SAD methodology
  - Homework's and deliverables tied to the project

Deliverables	Deadlines
Project Proposal	(3rd week)
System Requirement Specifications (SRS)	(6th week)
Progress Report 02	(9th week)
System Design Specifications (SDS)	(11th week)
Final Report/User Manual	(13th week)
Presentations and Demo	(15th -16th week)

Course Material

- You will have Presentations of each topic and ***reference books*** in PDF format will be available on slate.

Text Books

1. *UML 2 Toolkit by Hans-Erik Eriksson, Magnus Penker, Brian Lyons, David Fado*
2. *Applying UML and Patterns 3rd Edition by Craig Larman*

Reference Books

1. *UML and the Unified Process, Practical object-oriented analysis and design by Jim Arlow, Ila Neustadt*
2. *The Unified Modeling Language Reference Manual, 2nd edition by James Rumbaugh, Ivar Jacobson and Grady Booch*

- Hands-on labs will be part of the course.

Course Goals/Objectives

- By the end of this course, you will be able to
  - Perform analysis on a given domain and come up with a complete system analysis and Design (OOD).
  - Practice various techniques which are commonly used in analysis and design phases in the software industry.
  - Use Unified Modeling Language (UML) as a tool to demonstrate the analysis and design ideas
  - Analyze, design and implement practical systems of up to average complexity within a team and develop a software engineering mindset

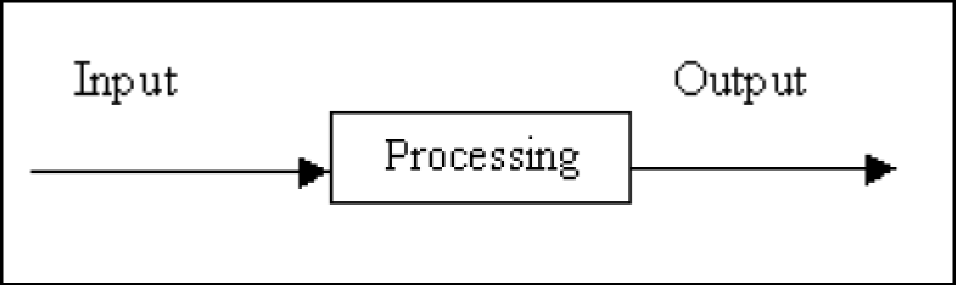
Why Study SDA?

What is System Analysis and Design?

- **What is a system?**
- **An organized way of dealing with a problem**
- **Analysis and Design of the system?**

System

- A collection of components that work together to realize some objective forms a system.
- Basically there are three major components in every system namely input, processing and output.



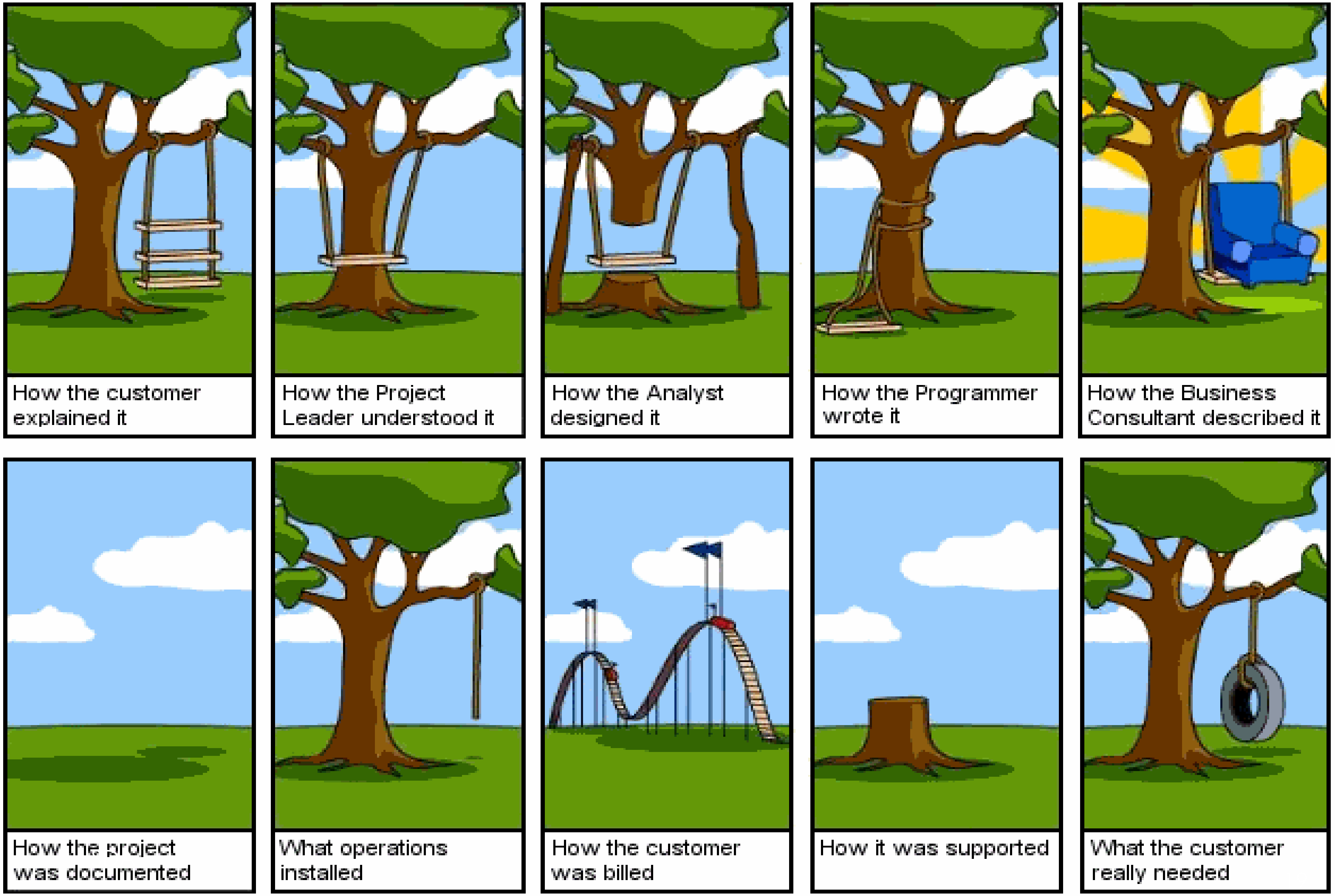
System

A System can be a Application program or it can be an Information System.

- Computer App: is an application program (app for short) is a computer program designed to perform a group of coordinated functions, tasks, or activities for the benefit of the user.
- Information System: is software that helps you organize and analyze data. This makes it possible to answer questions and solve problems relevant to the mission of an organization.

Defining the *problem* is the PROBLEM

Why Object-Oriented?


How the customer explained it
How the Project Leader understood it
How the Analyst designed it
How the Programmer wrote it
How the Business Consultant described it
How the project was documented
What operations installed
How the customer was billed
How it was supported
What the customer really needed

Software Crisis

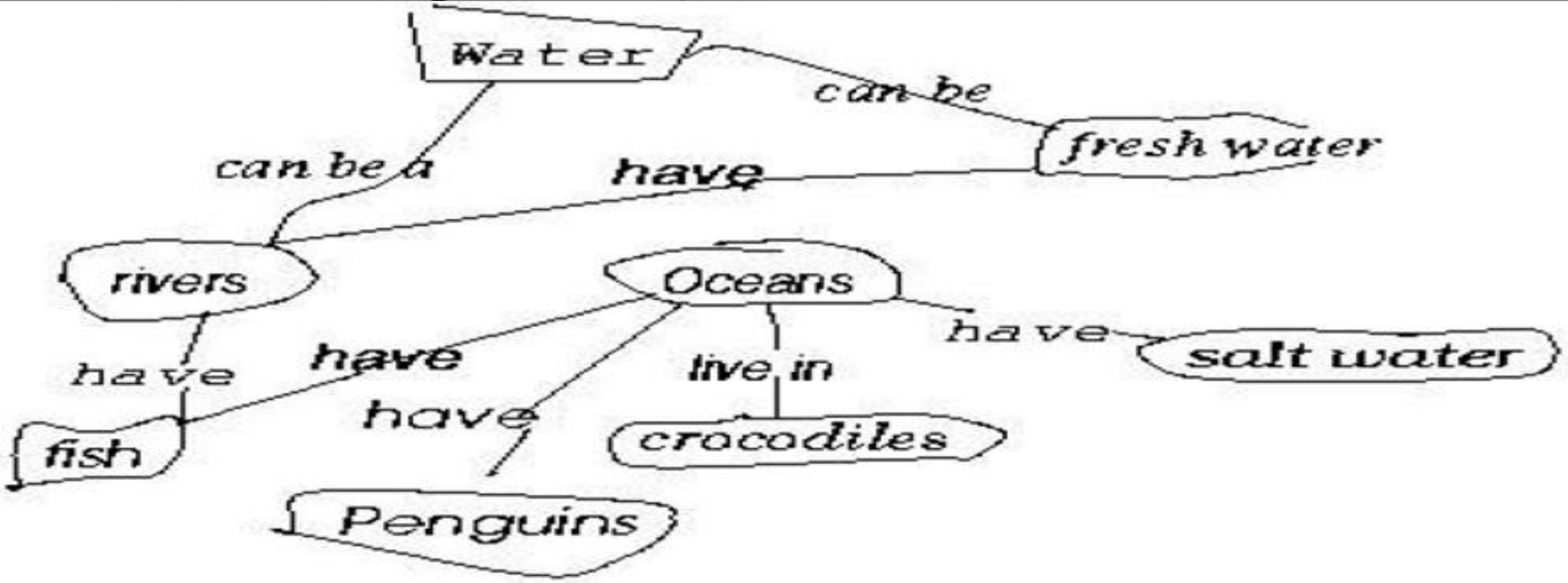
*"The "software crises" came about when people realized the major problems in software development were … caused by **communication** difficulties and the management of **complexity"** [Budd]*

What kind of language can alleviate difficulties with communication & complexity?

Why Object Oriented?

Study of a first grade class [Martin & Odell] [Novak, 1984, Cambridge University Press]

When given a list of concepts (water, salt water, Oceans, Penguins,...),
Harry constructed a concept diagram through which he understands his world and communicates meaning



Why Object-Oriented ->

What is a ***model*** **and why**?

- A model is a simplification of reality.

  E.g., a miniature bridge for a real bridge to be built
  - *Well…sort of….but not quite*
- A mental model is our simplification of our perception of reality
- A model is an *abstraction* of something for the purpose of *understanding*, be it the problem or a solution.

- **To understand *why* a software system is needed, *what* it should do, and *how* it should do it.**
- **To communicate our understanding of why, what and how.**
- **To detect commonalities and differences in your perception, my perception, his perception and her perception of reality.**
- **To detect misunderstandings and miscommunications.**

OO Analysis → OO Design → OO implementation

- The purpose of OO analysis and design can described as –
- Identifying the objects of a system.
- Identifying their relationships.
- Making a design, which can be converted to executables using OO languages.

Advantages of Object Oriented

- Simplicity
- Reusability
- Increase quality
- Faster development
- Maintainability
- Modifiability

Basic OOP Concepts and Terms

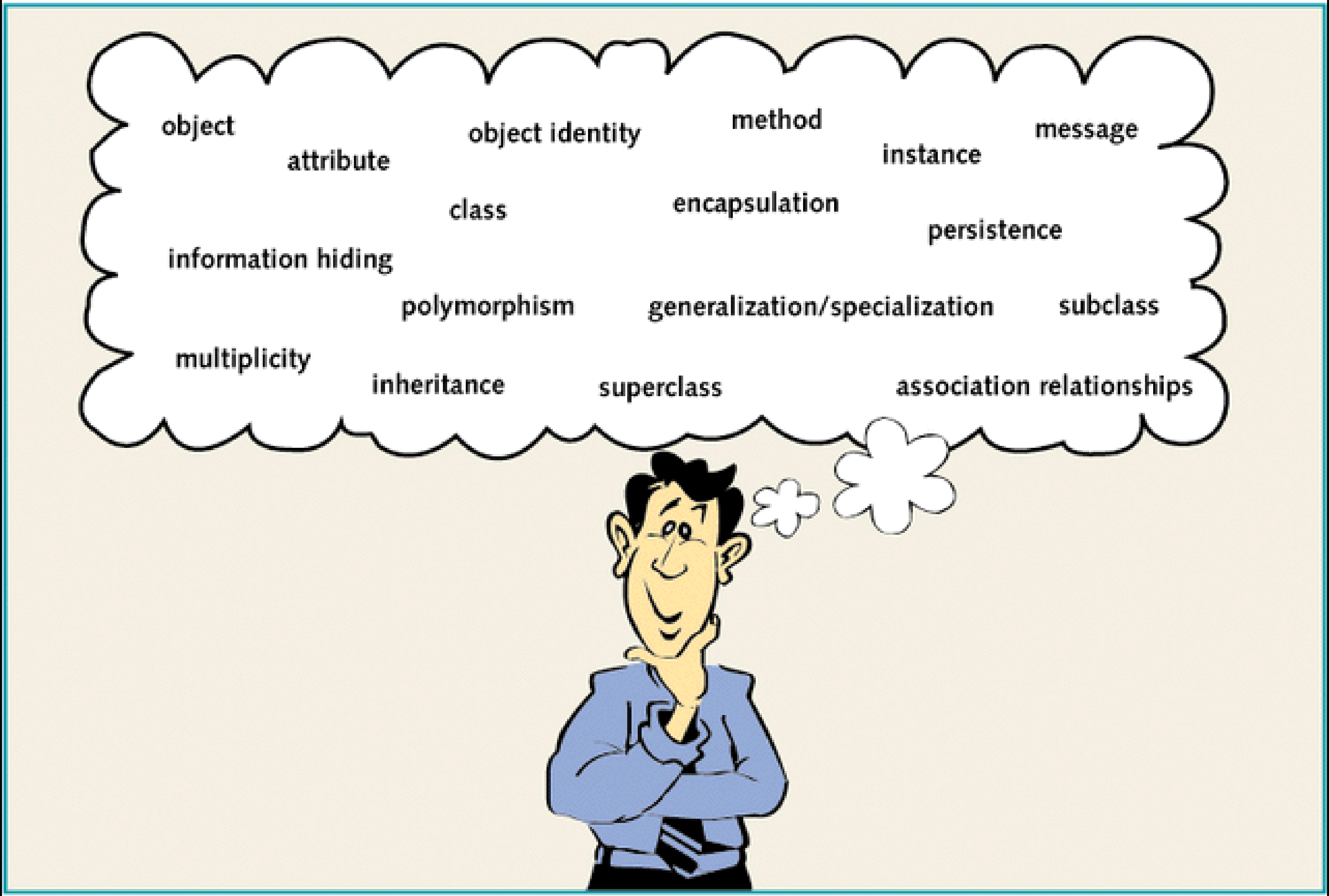


Figure 1-5 Key OO concepts

Objects

- Most basic component of OO design. Objects are designed to do a *small*, specific piece of work.
- Objects represent the various components of a business system

Examples of Different Object Types

- GUI objects
  - o objects that make up the user interface
  - o e.g. buttons, labels, windows, etc.
- Problem Domain objects
  - o Objects that represent a business application
  - o A problem domain is the scope of what the system to be built will solve. It is the business application.
    - e.g. An order-entry system
      A payroll system
      A student system

Sample Problem Domain

Company ABC needs to implement an order-entry system that takes orders from customers for a variety of products.

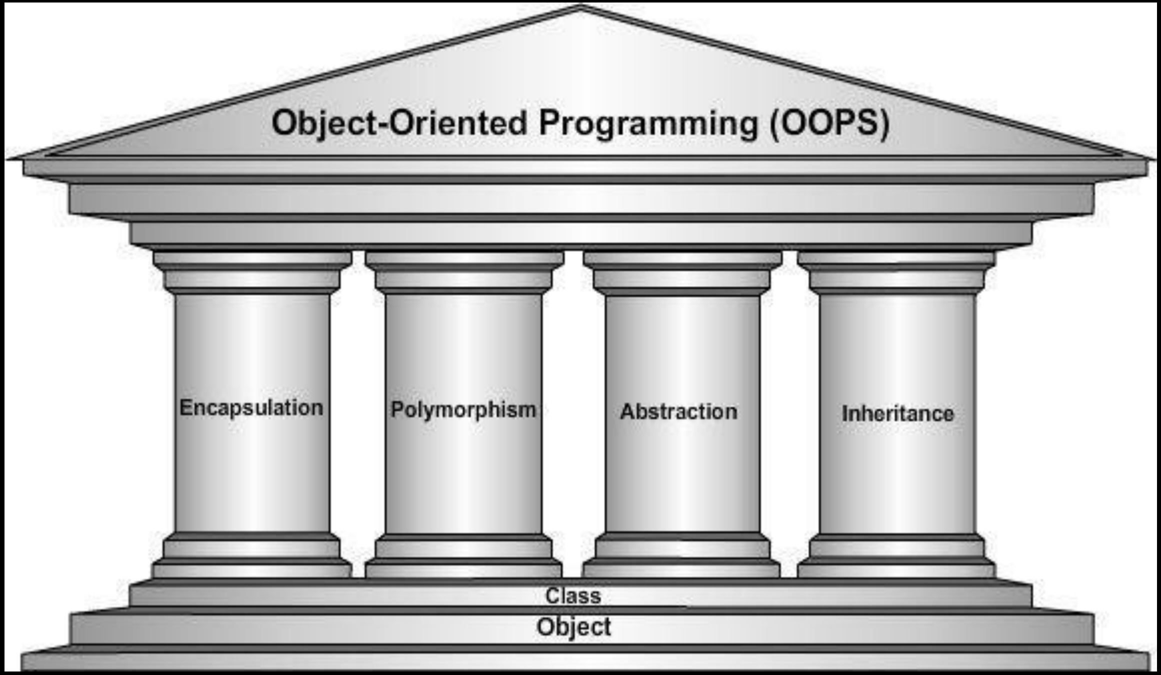
What are the objects in this problem domain?

Classes and Objects

- **Classes**
  - **Define what all objects of the class represent**
  - **It is like a blueprint. It describes what the objects look like**
  - **They are a way for programs to model the real world**
- **Objects**
  - **Are the instances of the class**

OO Principles

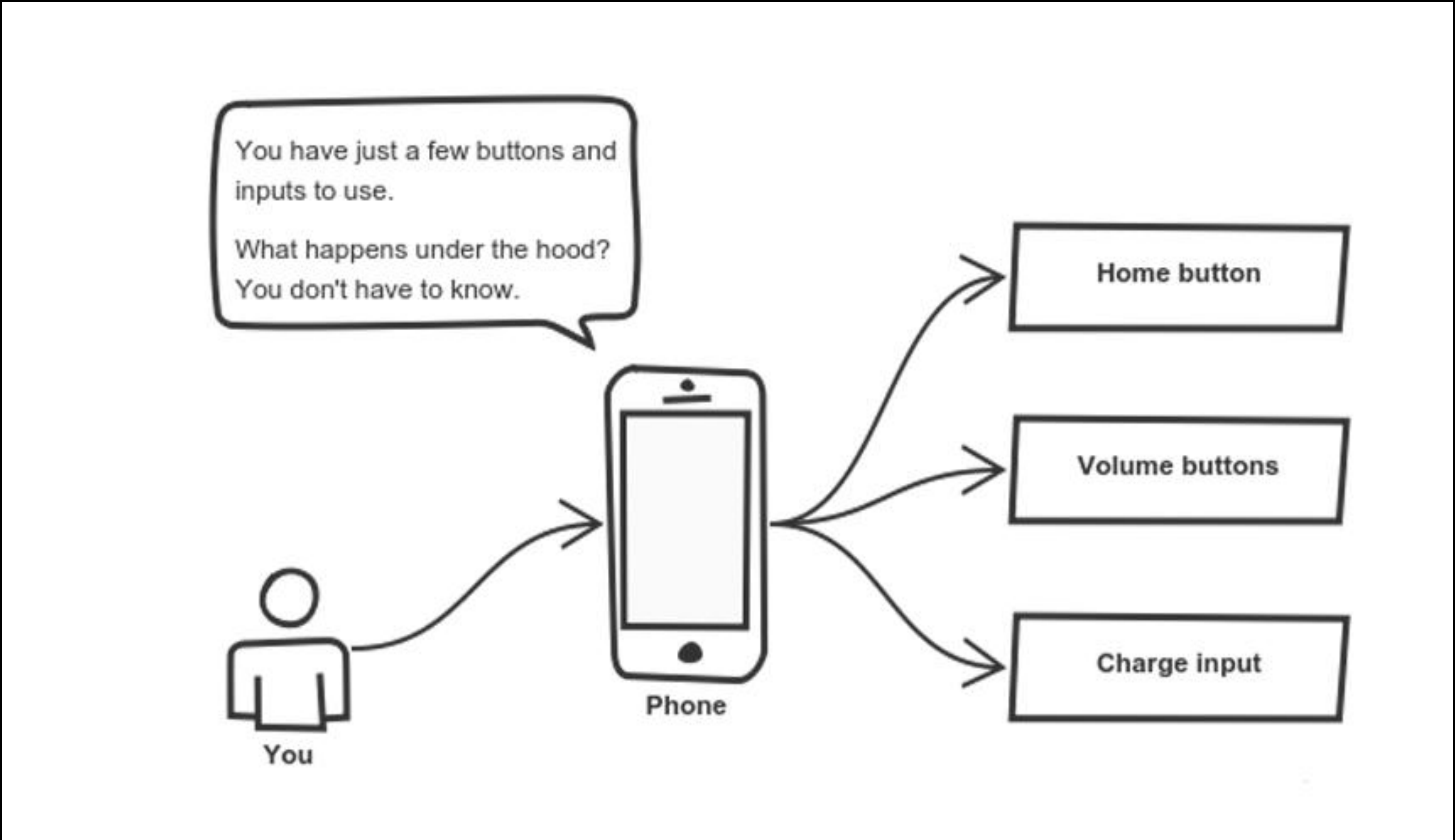
1) Abstraction
2) Encapsulation
3) Inheritance
4) Polymorphism



Principle 1: Abstraction

- Applying abstraction means that each object should only expose a high-level mechanism for using it.
- This mechanism should hide internal implementation details. It should only reveal operations relevant for the other objects

Principle 1: Abstraction


You have just a few buttons and inputs to use.
What happens under the hood? You don't have to know.
Home button
Volume buttons
Charge input
Phone
You

Cell phones are complex. But using them is simple.

Principle 2: Encapsulation

- Encapsulation separates implementation from users/clients.
- Objects have attributes and methods combined into one unit
- Encapsulation is achieved when each object keeps its state private, inside a class. Other objects don’t have direct access to this state. Instead, they can only call a list of public functions — called methods.

Principle 3: Inheritance

- One class of objects takes on characteristics of another class and extends them
- Superclass → subclass
- Generalization/specialization hierarchy
  - Also called an inheritance hierarchy
  - Result of extending class into more specific subclasses
- **This is an important concept!!**

Principle 4: Polymorphism

- **Polymorphism**
  - literally means “many forms”
  - in Java means using the same message (or method name) with different classes
    - different objects can respond in their own way to the same message

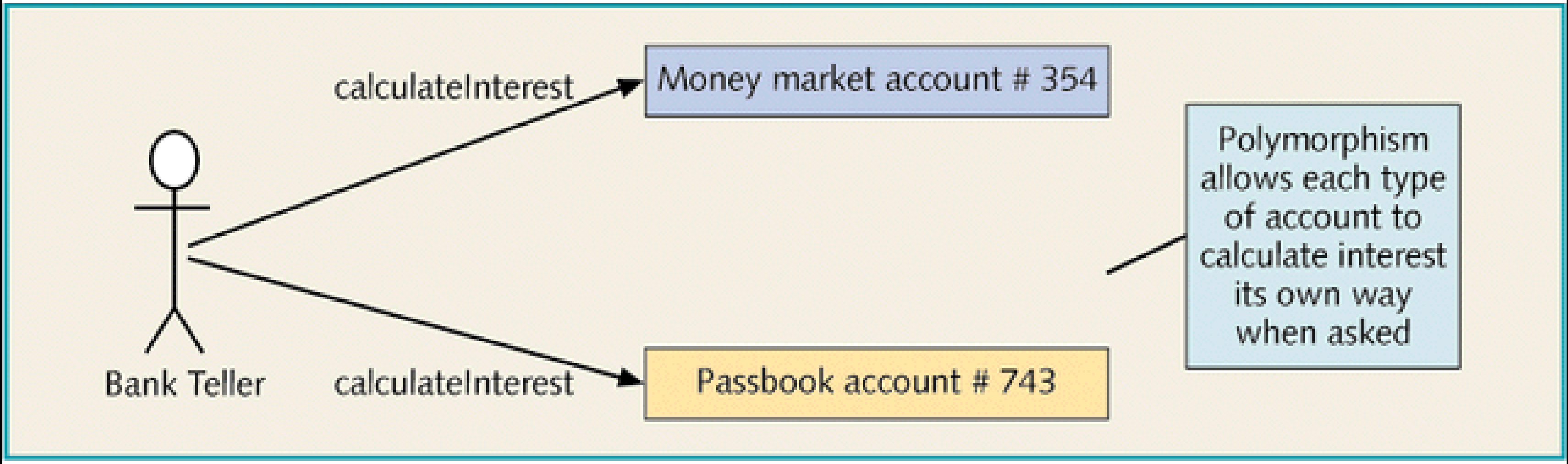
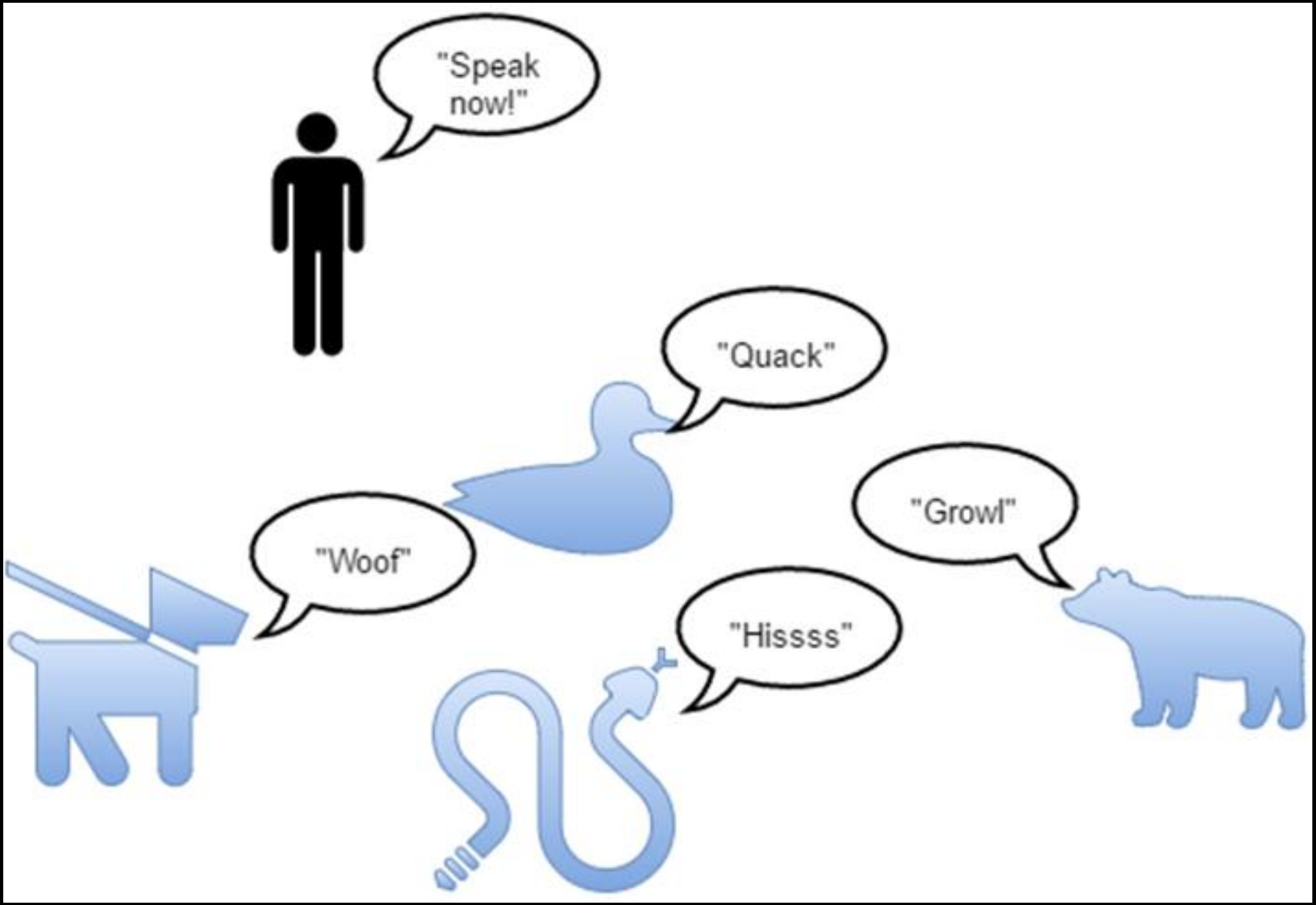

calculateInterest
Money market account # 354
Polymorphism allows each type of account to calculate interest its own way when asked
Bank Teller
calculateInterest
Passbook account # 743

Figure 1-12 Polymorphism for two different types of bank accounts

Polymorphism

- Objects of different classes respond to the same message differently.
- ability to dynamically choose the method for an operation at run-time or service-time



Today's Outline

- Software Development methodologies
  - SAD
  - OOAD
- SAD vs. OOAD
- Case Study
- Software development Categories

Structured Analysis and Design

What is the SAD?

Divide and Conquer

Traditional systems development technique that is time tested and easy to understand

Uses set of process models to describe a system graphically

Divide large, complex problem into smaller, more easily handled ones.

Top Down Approach

Functional view of the problem

Structured Analysis and Design

What is the SAD?

Establish complete requirement documentation

Establish concrete requirement specification

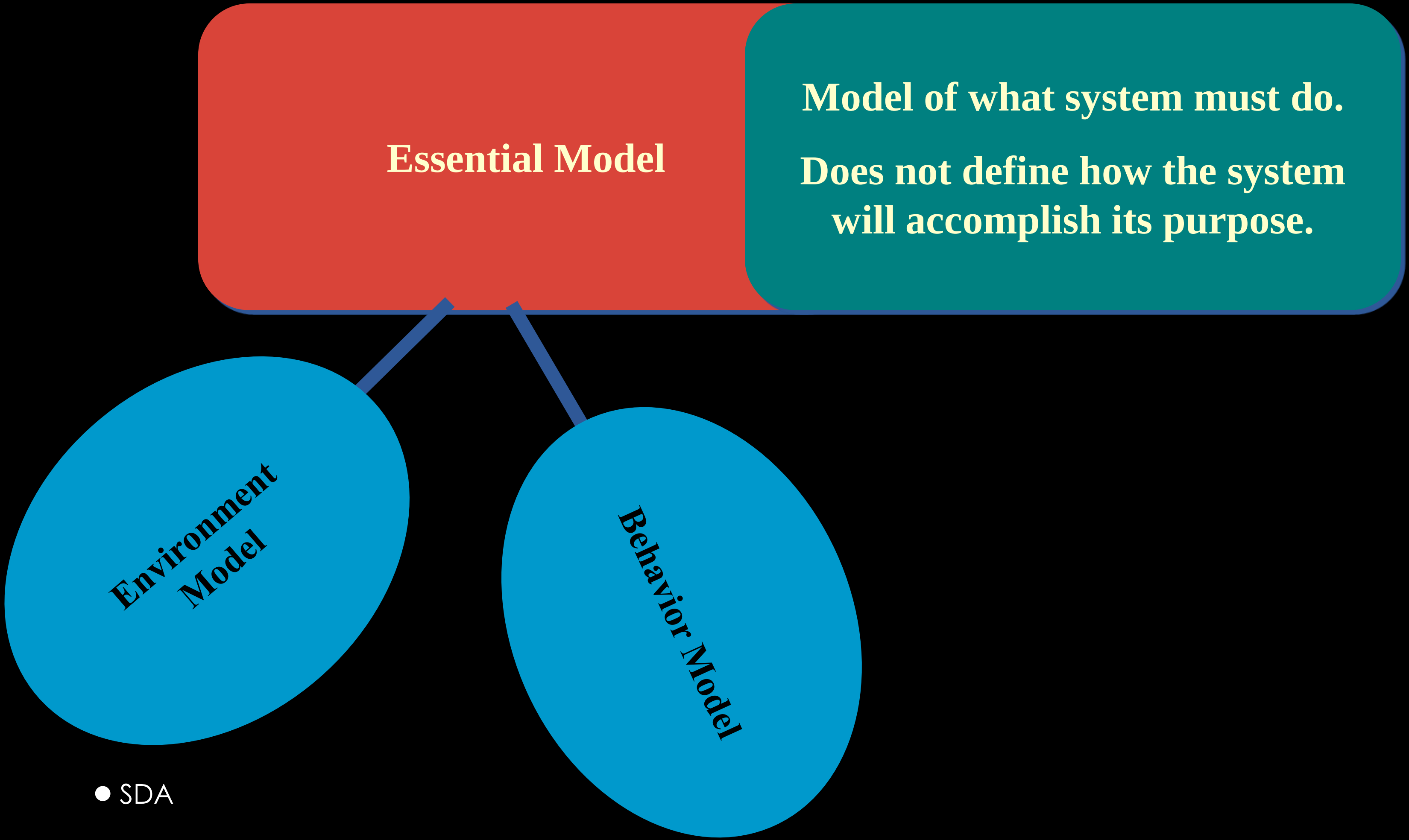
Structured analysis is a set of techniques and graphical tools that allow the analyst to develop a new kind of system specification that are easily understandable to the user. Analysts work primarily with their wits, pencil and paper."

Improve Quality and reduce risk

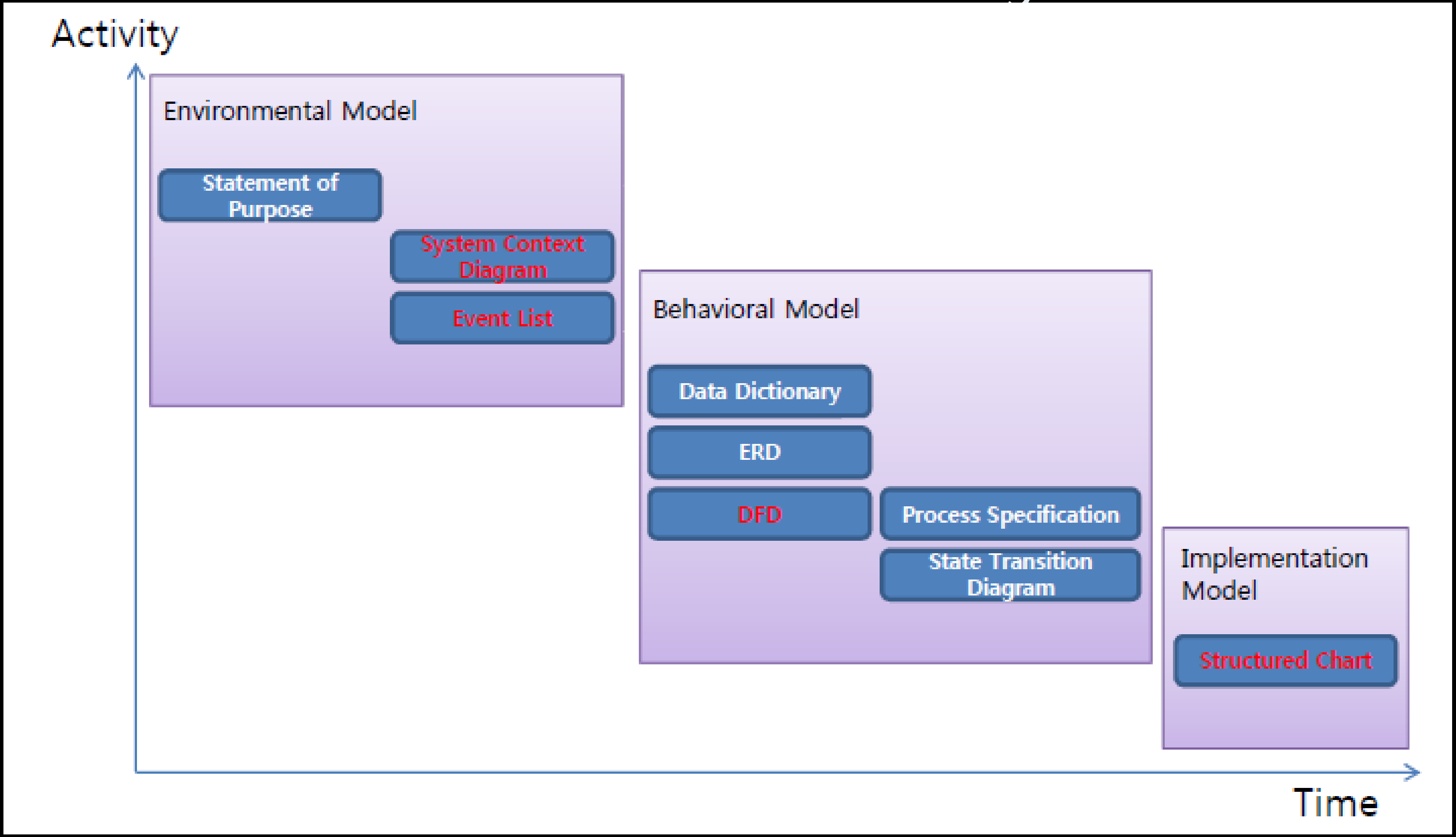
Focus on reliability, flexibility & maintainability

Structured Analysis and Design

Elements of Structured Analysis and Design?


Essential Model
Model of what system must do.
Does not define how the system will accomplish its purpose.
Environment Model
Behavior Model

SAD Activity



Structured Analysis and Design

Advantages

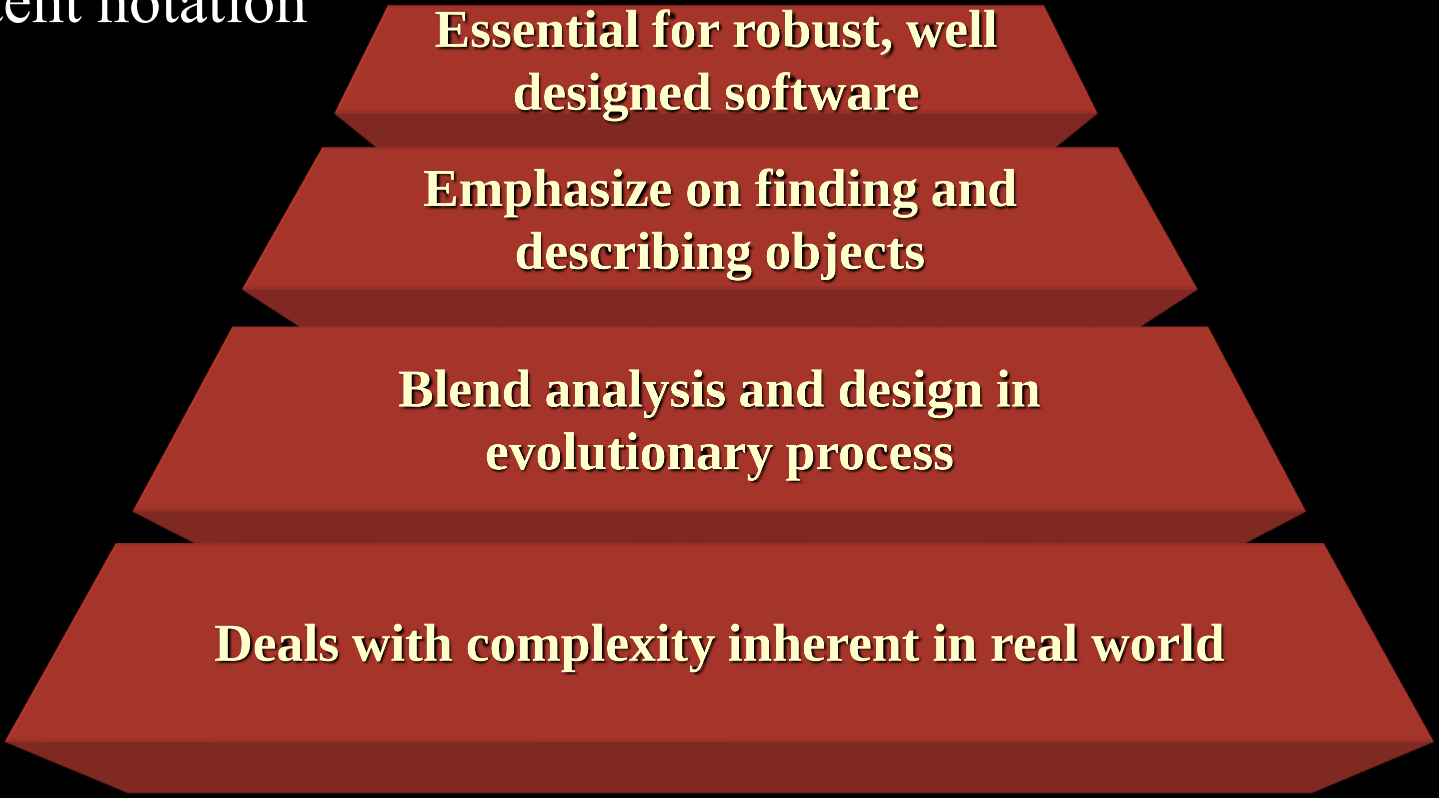
- Visual, so it is easier for users/programmers to understand
- Makes good use of graphical tools
- A mature technique
- Process-oriented approach is a natural way of thinking
- Flexible
- Simple and easy to understand and implement

Disadvantages

- Not enough user-analyst interaction
- It depends on dividing system to sub systems but it is to decide when to stop decomposing.

Object Oriented Analysis and Design

Object-Oriented analysis and design thoroughly represent complex relationships, as well as represent data and data processing with a consistent notation


Essential for robust, well designed software
Emphasize on finding and describing objects
Blend analysis and design in evolutionary process
Deals with complexity inherent in real world

Object Oriented Analysis and Design

Analysis Phase

- Model of the real-world application is developed showing its important properties.
- Model specifies the functional behavior of the system independent of implementation details

Design Phase

- Analysis model is refined and adapted to the environment.
- System design: Concerned with overall system architecture
- Object design: Implementation details are added to system design

Implementation Phase

- Design is implemented using a programming language or database management system

Object Oriented Analysis and Design

From Analysis to Implementation

Analysis

Investigation of the problem

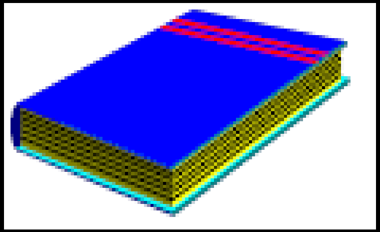
Design

Logical Solution

Construction

Code

Domain Concept
Ex: Book (Concept)



Logical Software Objects

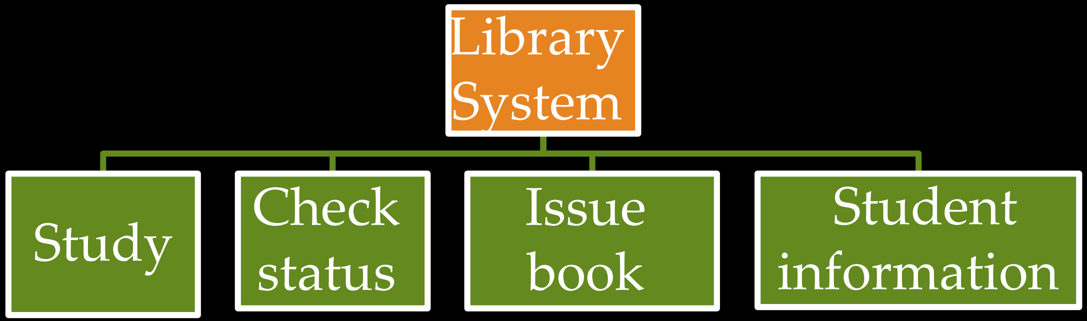
Book
Attribute: Title
Method: Display()

Representation in an OO Programming Language

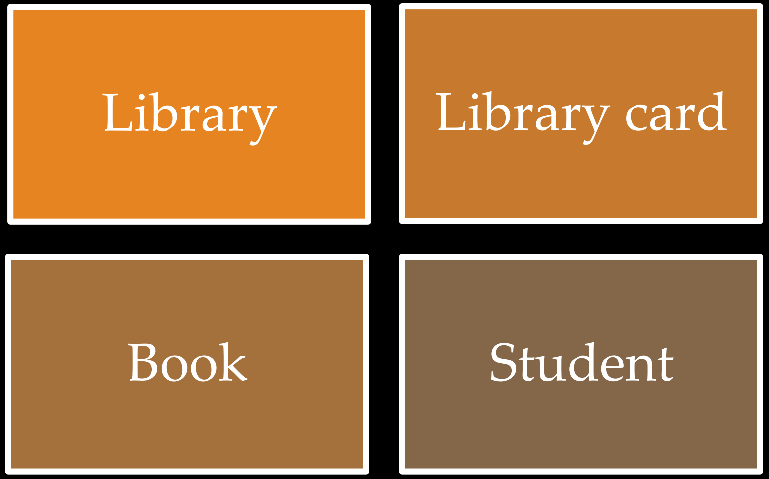
```
Public Class Book {
  Private String Title;
  Public void
  Display();
}
```

Similarities and difference between SAD and OOAD

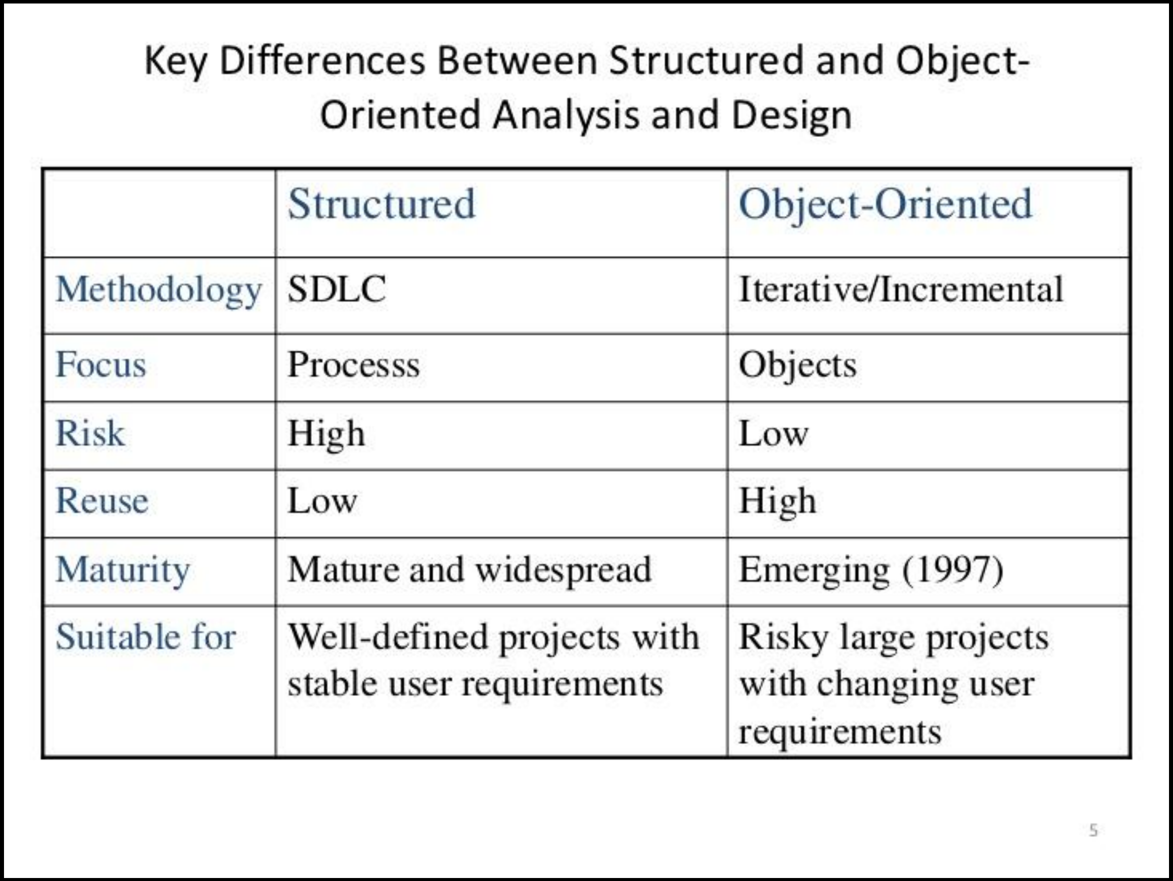
SAD



OOAD



Similarities and difference between SAD and OOAD



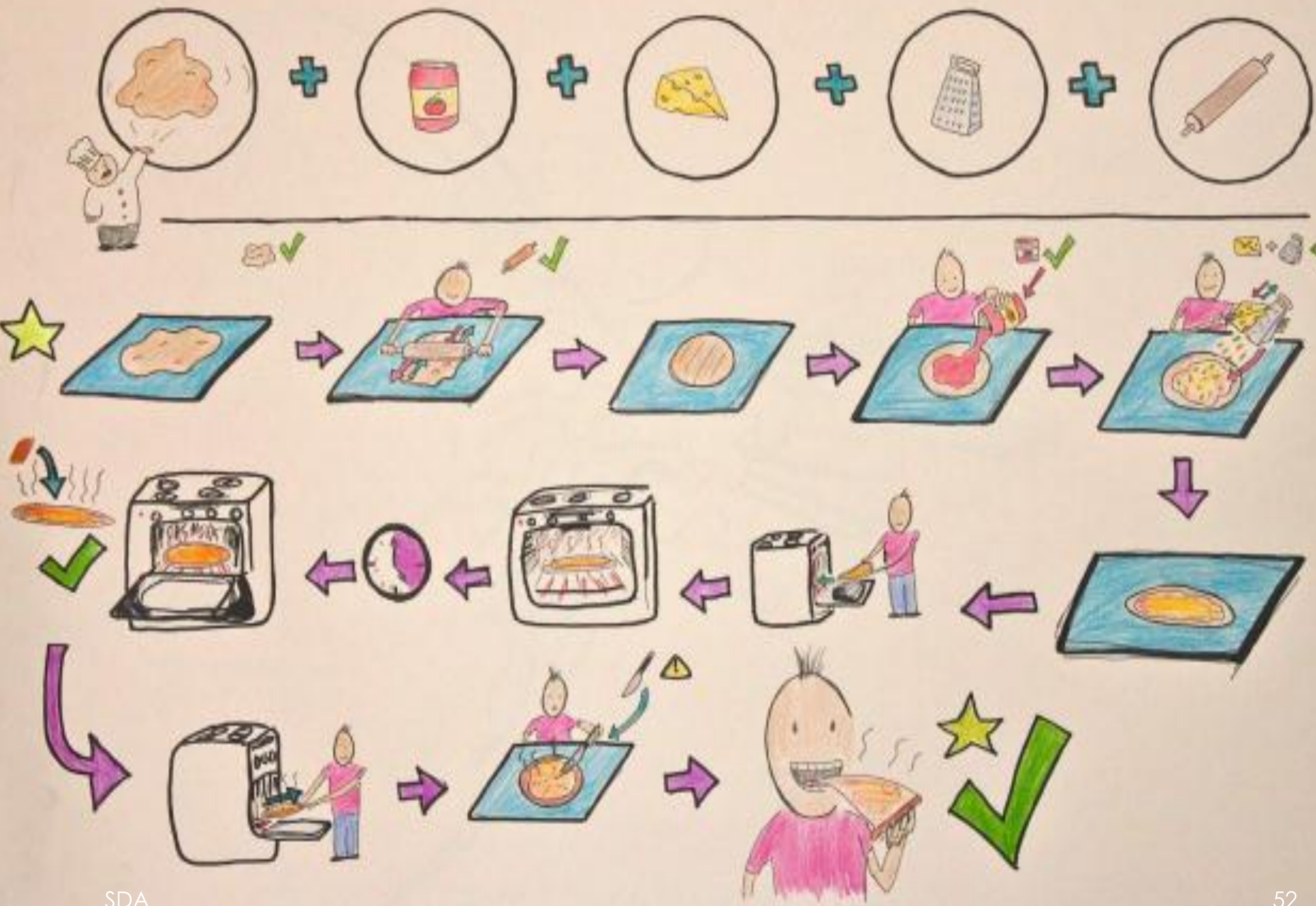
Key Differences Between Structured and Object-Oriented Analysis and Design

	Structured	Object-Oriented
Methodology	SDLC	Iterative/Incremental
Focus	Processs	Objects
Risk	High	Low
Reuse	Low	High
Maturity	Mature and widespread	Emerging (1997)
Suitable for	Well-defined projects with stable user requirements	Risky large projects with changing user requirements

5

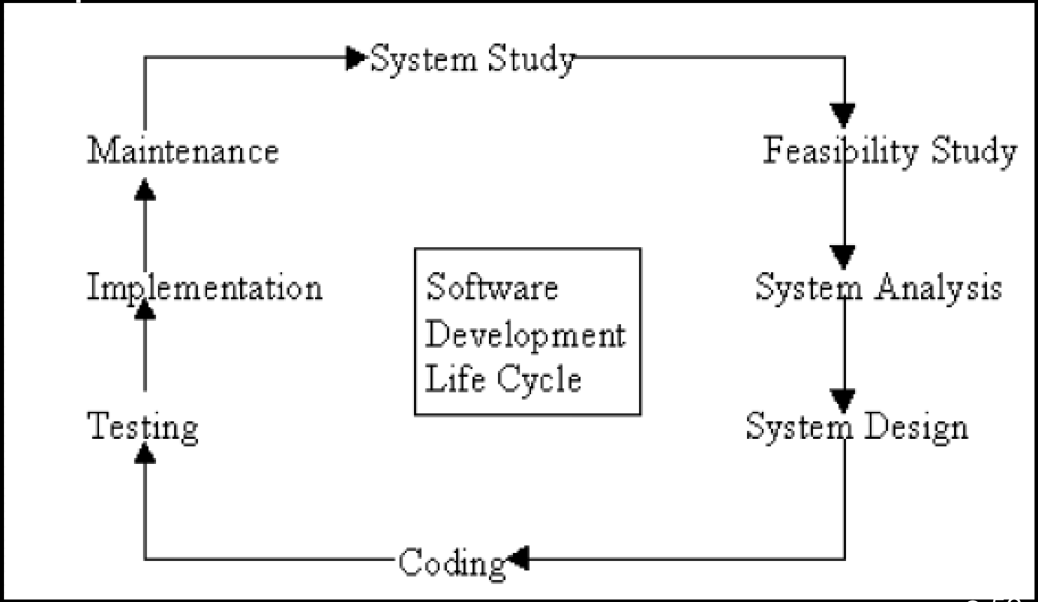
Today's Outline

- What is system?
- SDLC
- SDLC Phases

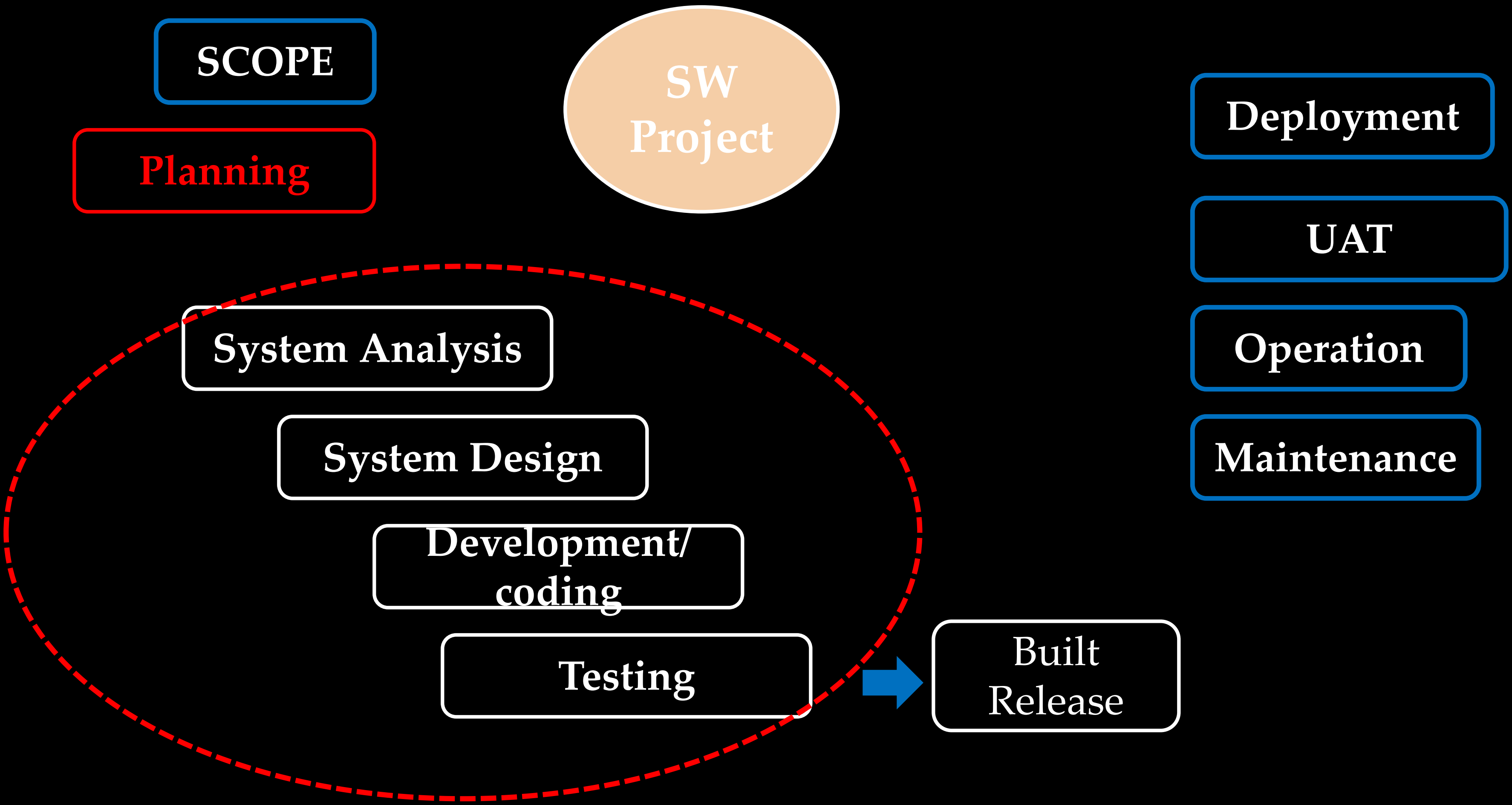


SDLC

- Software Development Life Cycle (SDLC) is a process used by the software industry to design, develop and test high quality software.
- It consists of a detailed plan describing how to develop and maintain software.
- SDLC consists of many activities/ phases.
- Following are the major phases of SDLC.
  - System study
  - Feasibility study
  - System analysis
  - System design
  - Coding
  - Testing
  - Built release
  - Maintenance



Software Development Life Cycle



The System Development Life Cycle

What are guidelines for system development?

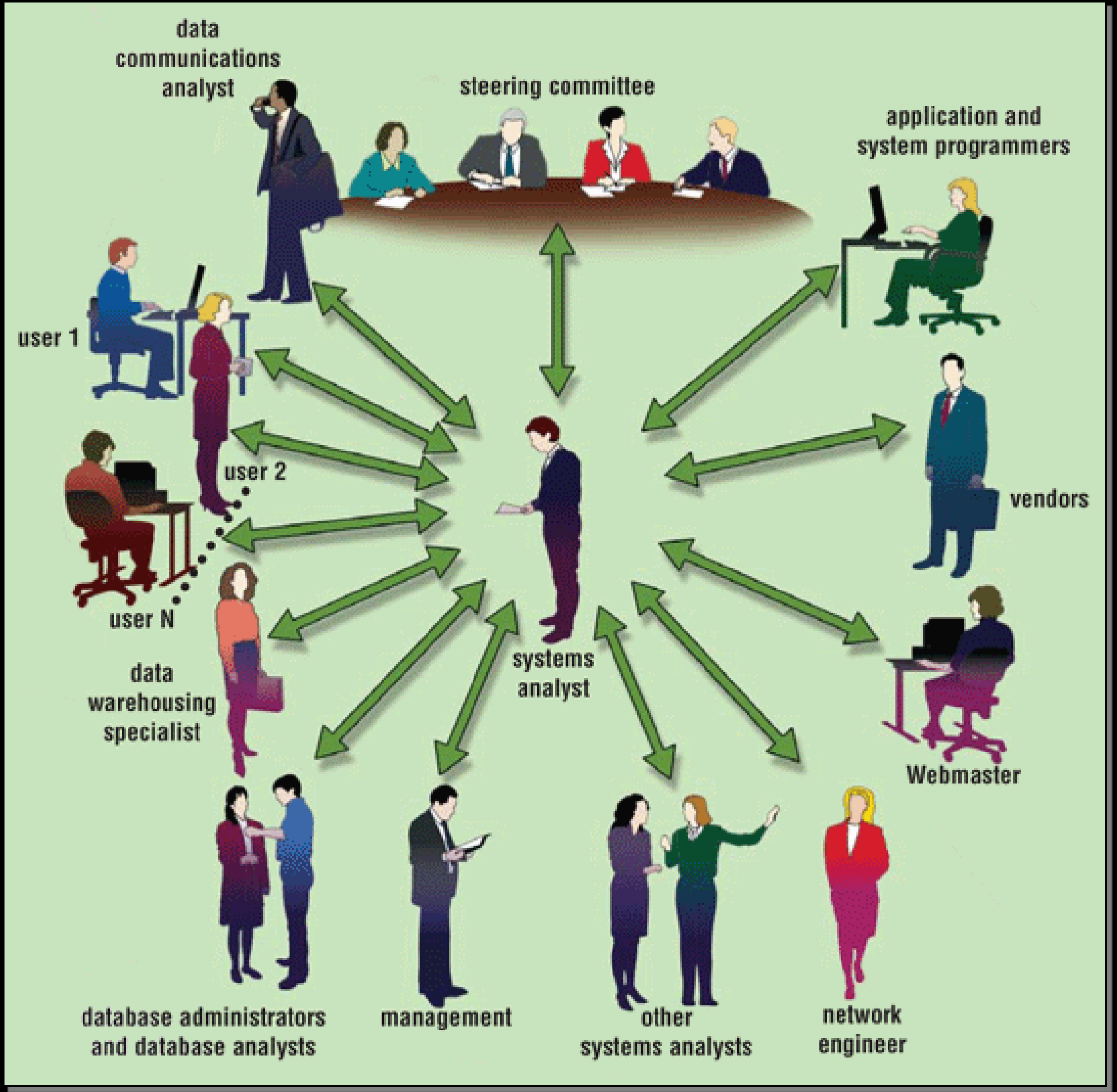
Arrange tasks into phases (groups of activities)

Involve users (anyone for whom system is being built)

Develop clearly defined standards (procedures company expects employees to follow)

The System Development Life Cycle

Who participates in the system development life cycle?


data communications analyst
steering committee
application and system programmers
user 1
user 2
user N
vendors
systems analyst
data warehousing specialist
Webmaster
database administrators and database analysts
management
other systems analysts
network engineer

The System Development Life Cycle

What is the project team?

Formed to work on project from beginning to end

Consists of users, systems analyst, and other IT professionals

Project leader—one member of the team who manages and controls project budget and schedule

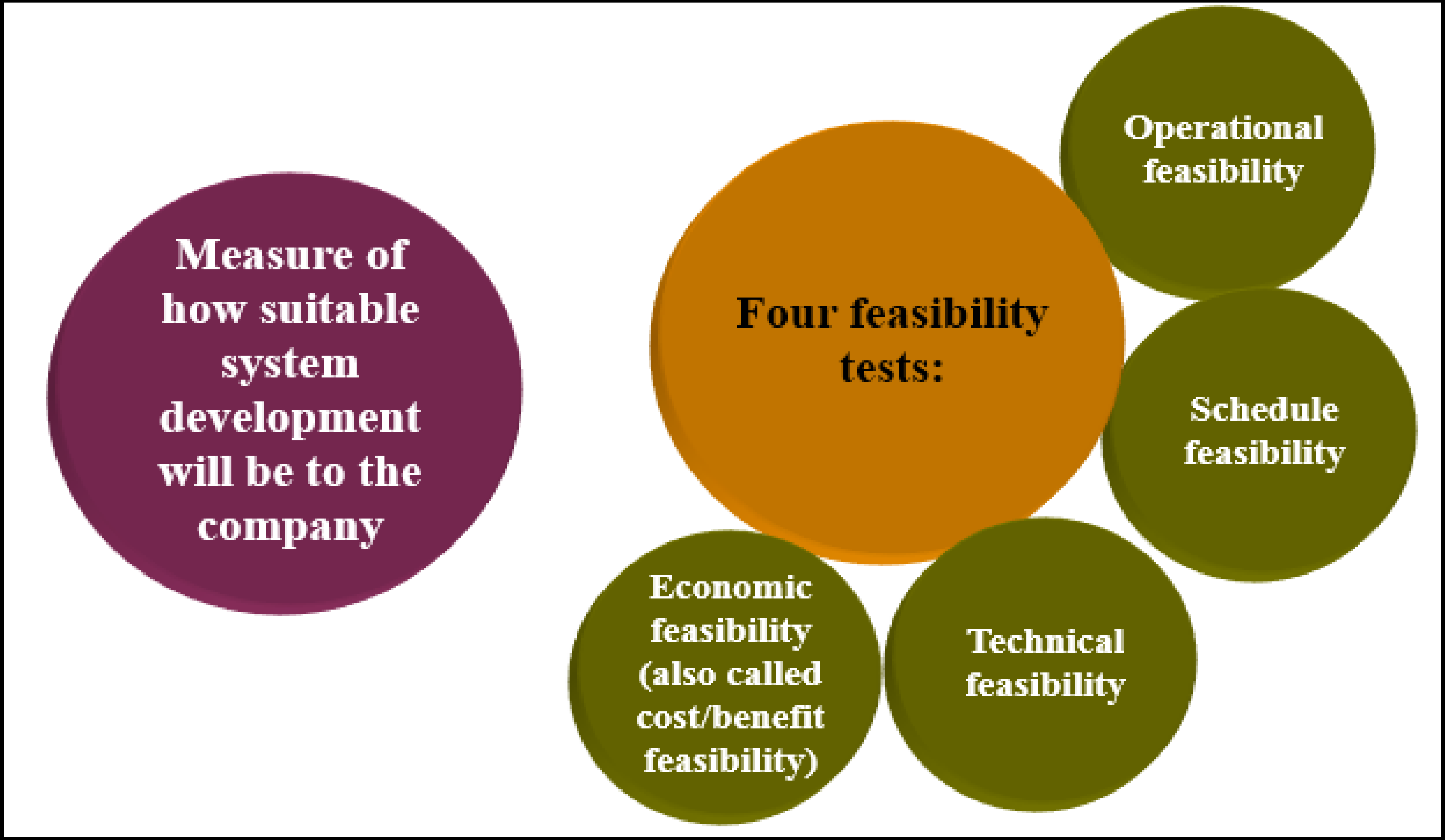
Stage 1: System Study

- Gives clear picture of what actually the physical system is.
- System study phases(I & II):
  - I: initial survey of the system(scope identification)
  - II: depth study of the system (requirement identification, limitation & issues of the current system). It also includes the back ground analysis and inference or findings of the system.
- Output: system proposal or recommendations to overcome the limitations / issues of the current system.

Stage 2: Feasibility Study

- A feasibility study precedes the decision to begin a project. It is an assessment of the practicality of a proposed system.
- Three main types of feasibility study:
  - **Technical feasibility:** This assessment is based on an outline design of system requirements, to determine whether the company has the technical expertise to handle completion of the project.
  - **Operational feasibility:** refers to the measure of solving problems with the help of a new proposed system. It helps in taking advantage of the opportunities and fulfills the requirements as identified during the development of the project. It takes care that the management and the users support the project.
  - Economical feasibility: A project is considered economically feasible when the benefits that will accrue to the broad community are greater than the cost of undertaking the project.
- A feasibility study leads to a decision: go or no-go.
- Output: FSR

What is feasibility?


Measure of how suitable system development will be to the company
Four feasibility tests:
Operational feasibility
Schedule feasibility
Economic feasibility (also called cost/benefit feasibility)
Technical feasibility

Stage 3: System Analysis

- Forms the basis of agreement between user and developer. System analysis establish the system's services, constraints and goals by consultation with users.
- It is the study of specifications, operations and relationships with in the system and outside the system.
- Specifies what not how. (Hard task)
- Define the boundary of the new system keeping in view the problems and the new requirement.
- Output: is the Software Requirements Specification (SRS) document.

Stage 4: System Design

- A major step in moving from *problem* to *solution*.
- Based on system analysis, the new system must be designed.
- *Two main tasks*
  - *General design:* (preliminary design) components and connectors that should be there in the system
  - *Detailed design:* *(Detailed Design)* logic of modules

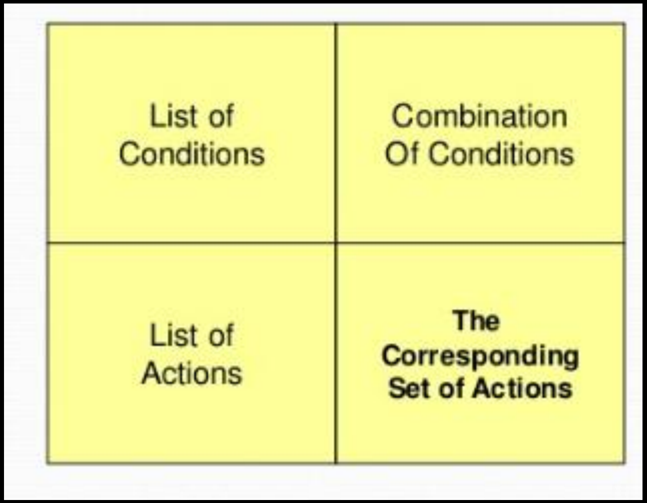
Output: SDS(system Design Specification)

Tools and Techniques for Designing

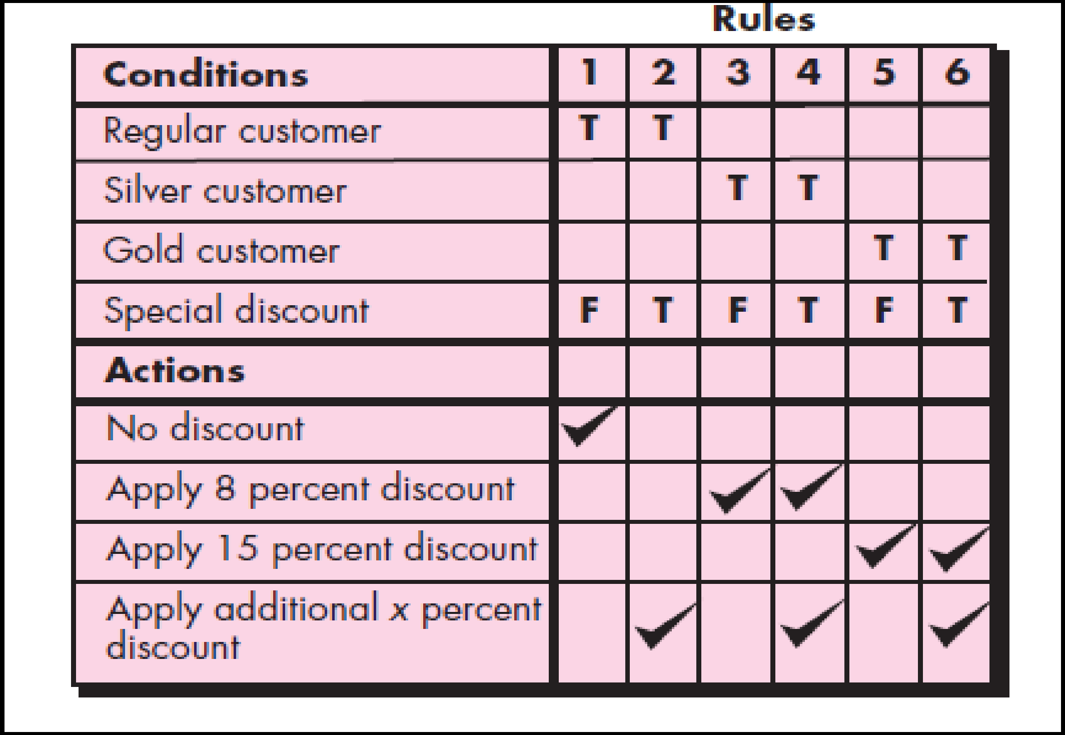
1. Flow Chart
2. Data Flow Diagram
3. Data Dictionary
4. Structured English
5. Decision Tables

Decision Tables

1) List all <u>actions</u> that can be associated with a specific procedure (or module)
2) List all <u>conditions</u> (or decisions made) during execution of the procedure



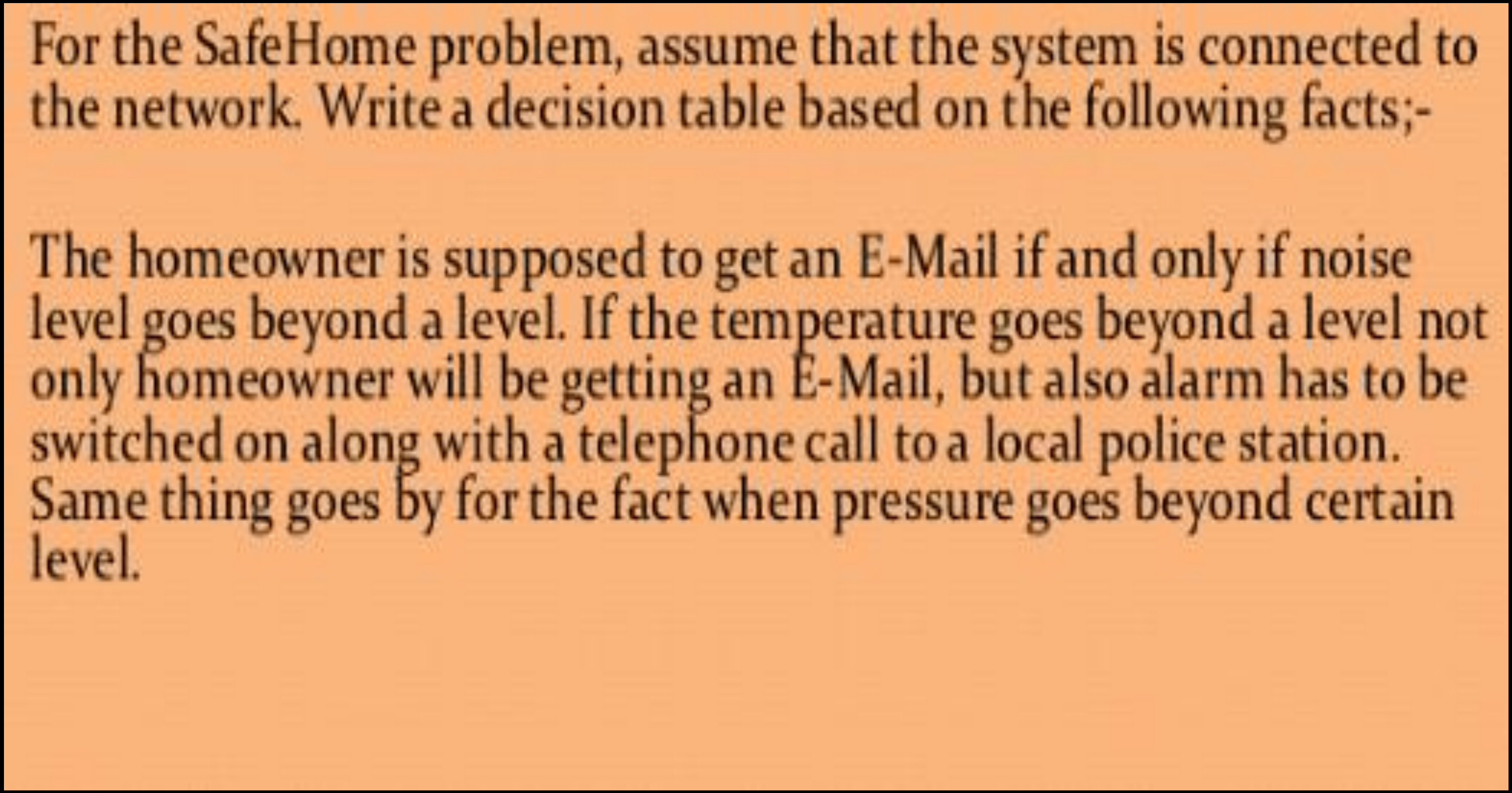
List of Conditions	Combination Of Conditions
List of Actions	**The Corresponding Set of Actions**



	Rules					
Conditions	**1**	**2**	**3**	**4**	**5**	**6**
Regular customer	T	T				
Silver customer			T	T		
Gold customer					T	T
Special discount	F	T	F	T	F	T
Actions						
No discount	✓					
Apply 8 percent discount			✓	✓		
Apply 15 percent discount					✓	✓
Apply additional *x* percent discount		✓		✓		✓

Decision Tables

Activity Task:



For the SafeHome problem, assume that the system is connected to the network. Write a decision table based on the following facts;-

The homeowner is supposed to get an E-Mail if and only if noise level goes beyond a level. If the temperature goes beyond a level not only homeowner will be getting an E-Mail, but also alarm has to be switched on along with a telephone call to a local police station. Same thing goes by for the fact when pressure goes beyond certain level.

Stage 5: Coding

- Converts design into ***code*** in specific language
- ***Goal:*** Implement the design with simple and easy to understand code
- programs must be modular in nature. This helps in fast development, maintenance and future changes, if required.
- Coding phase affects both ***testing*** and ***maintenance***.
    - Well written code reduces testing and maintenance effort.
- ***Output:*** is source-code.

Stage 6: Testing

- ***Defects*** are introduced in each phase
  - Must be found and removed to achieve high quality
- Software testing is a process of analyzing software for the purpose of finding bugs.
- Using test data, following test runs are carried out
  - Unit test: performed by the respective developers on the individual units of source code to ensure that the individual parts are correct in terms of requirements and functionality.
  - System test: done after unit test. System testing tests the system as a whole. Actual output of the system is matched with the expected outputs. Errors are identified and fixed.
  - *User acceptance testing (UAT)* – determines if the system satisfies the business requirements

- ***Outputs:*** are
  - Test plans/results
  - Final tested (reliable) code

Stage 7: Built Release

- After UAT, deployment phase begins.
- Final phase of SDLC, puts the product into production
- All programs of system are loaded onto the user's computer.
- Then training of user starts including
    - how to execute the package
    - how to enter the data
    - how to process data

Built Release Strategies

- Parallel run: computerized & manual systems are executed in parallel.
- Advantages of Parallel run:
  - Manual results comparison with the computerized one.
  - Failure of the computerized system at the early stage, does not affect the working of the organization.
- Pilot run: New system is installed in parts. Some part of the new system is installed first and executed successfully for considerable time period.
- Advantages:
  - When results are found satisfactory then only other parts are implemented.
  - This strategy builds the confidence and the errors are traced easily.

Stage 8: Maintenance

- Maintenance phase focuses on changes that associated with
  - Error Correction
  - Platform Adaptations required
  - Enhancement due to change
  - Re-engineering
- Maintenance is required to:
  - eliminate errors in the system during its working life
  - tune the system to any variations in its working environment.
- System Review: is necessary from time to time for:
  - knowing the full capabilities of the system
  - knowing the required changes or the additional requirements
  - studying the performance
- Major change during the review:
  - If a major change to a system is needed, a new project may have to be set up to carry out the change.
  - New project will then proceed through all above life cycle phases.

System Environments

- Development
- Test
- Staging
- Pre-Production
- Production
- Mirror
- Roles involved:
- D, PM, TM, BA.

That is all