System Design and Analysis
CS-324

Introduction

Lecture # 01,02
31 Aug, 1, 3 Sep

Rubab Jaffar
rubab.jaffar@nu.edu.pk

Today's Outline

- Administrative Stuff
- Overview of 324
- Introduction to system analysis and design
- Why study A&D?

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)

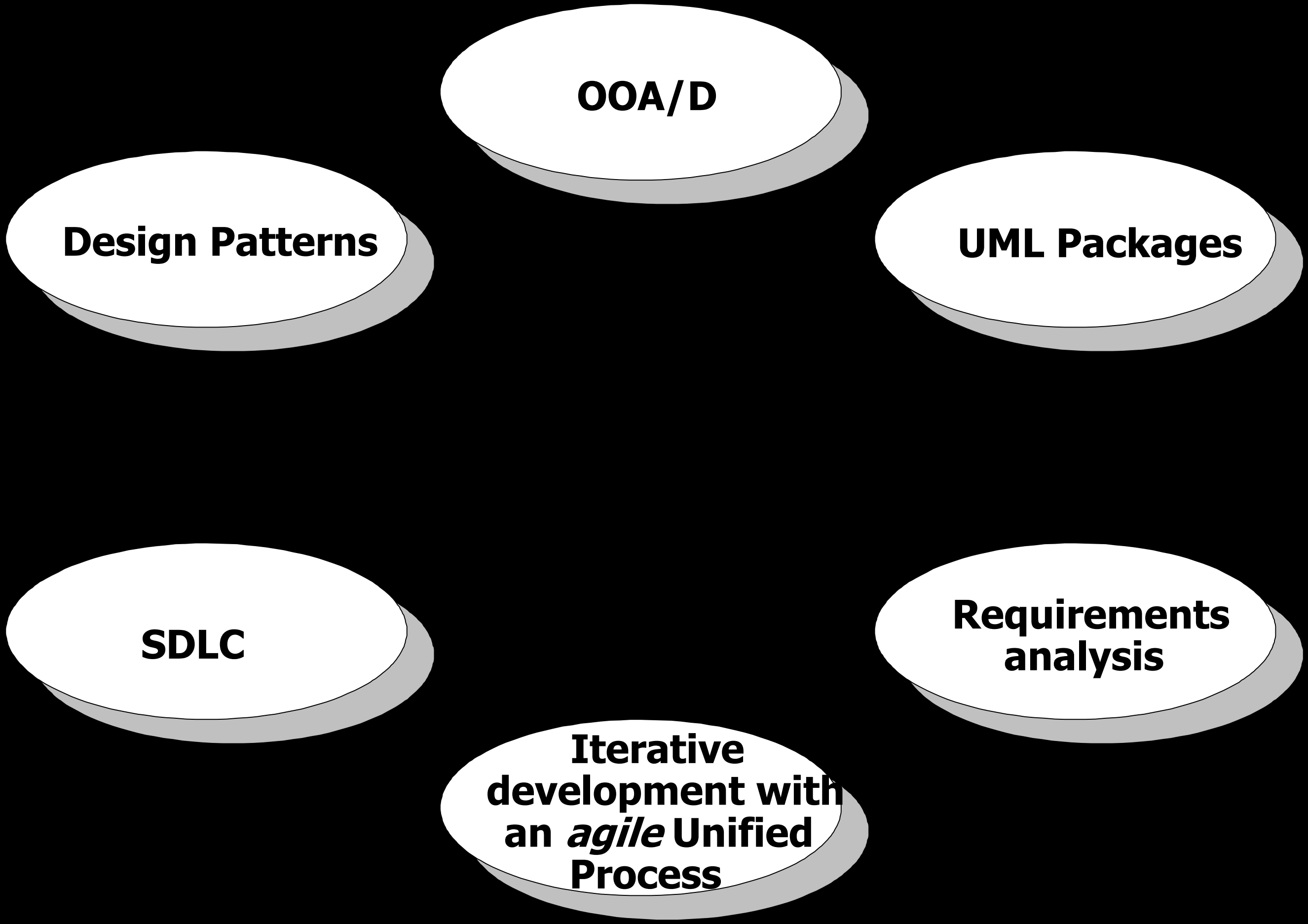
Overview of 324

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 324

Major Topics of the course



Overview of 324

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
  - o can design, implement, test, debug, read, understand and document the programs.

Tentative Grading Policy

- Assignments/Class Participation: 10 %
- Quizzes: 10%
- Mid Exam 1: 15 %
- Mid Exam 2: 15 %
- Final: 40%
- 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*
3. *Software Engineering Ian Sommerville 10 edition*

- 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 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

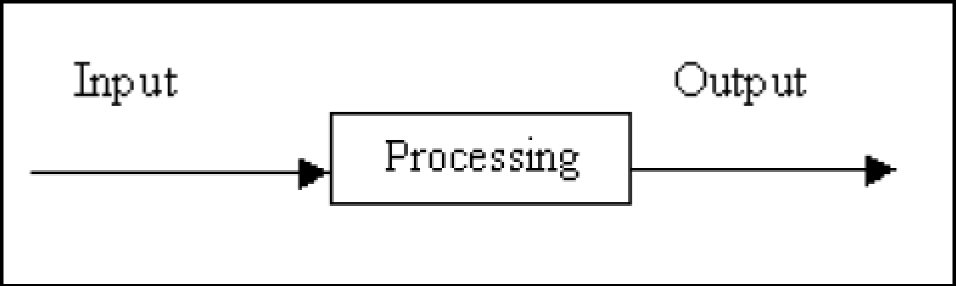
Why Study SAD?

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.

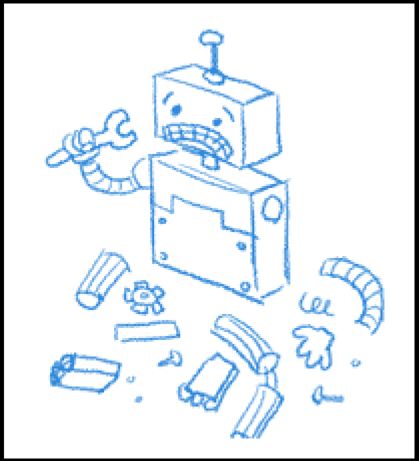


System

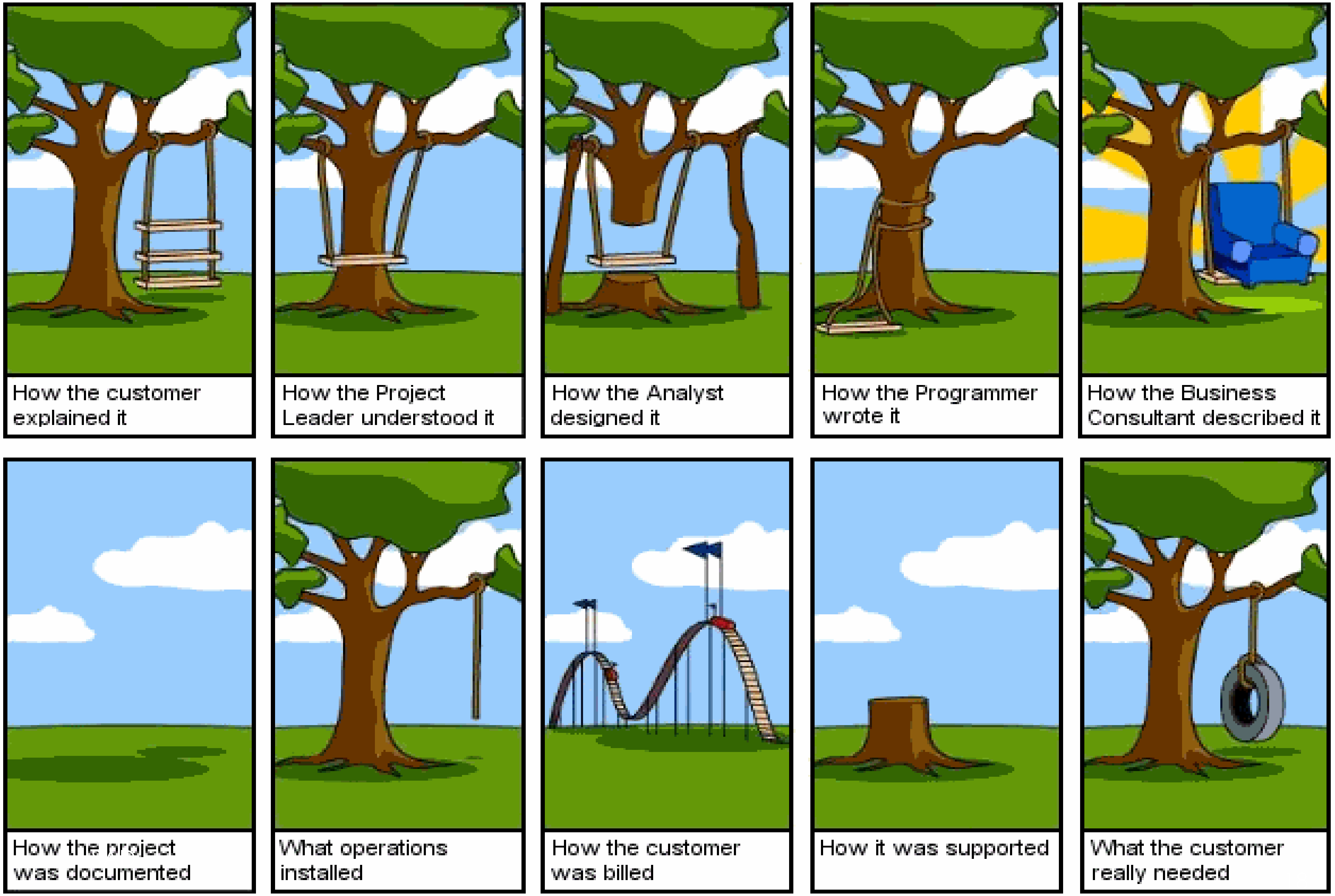
A System can be a Application program or it can be an Information System.

- 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

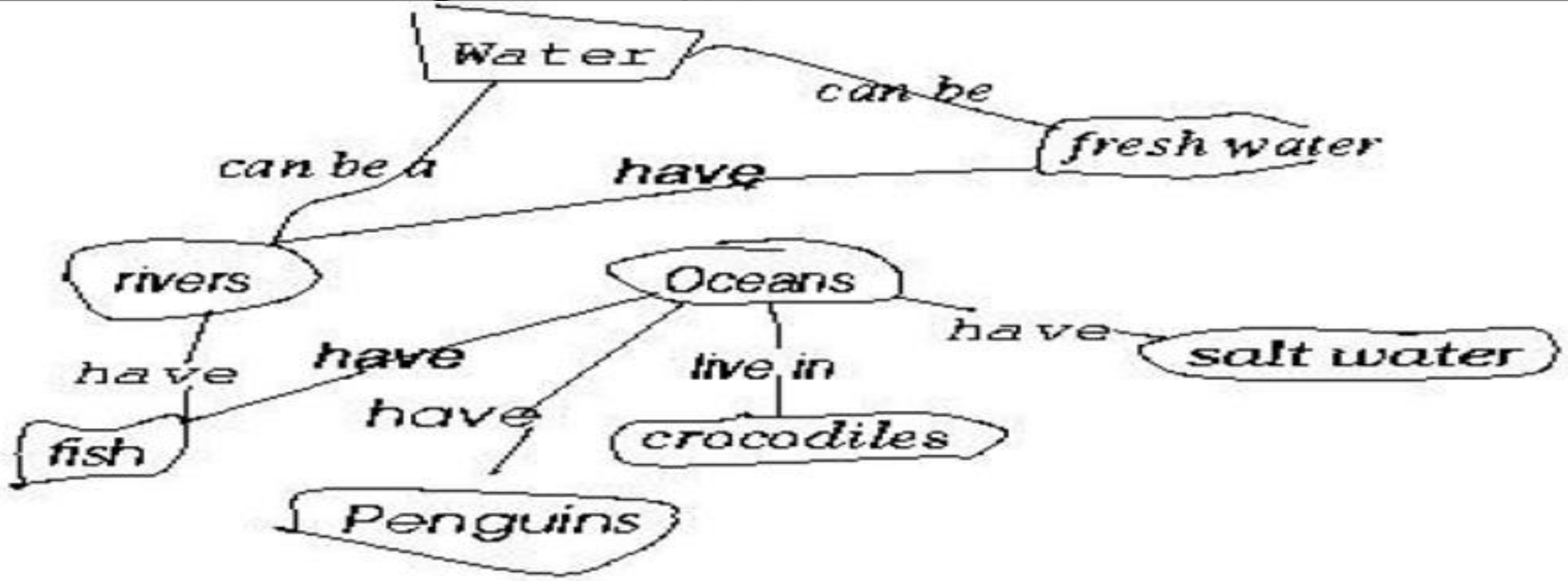
*"The "software crises" came about when people realized the major problems in software development were … caused by **communication** difficulties and the management of **complexity"** [Budd]*

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

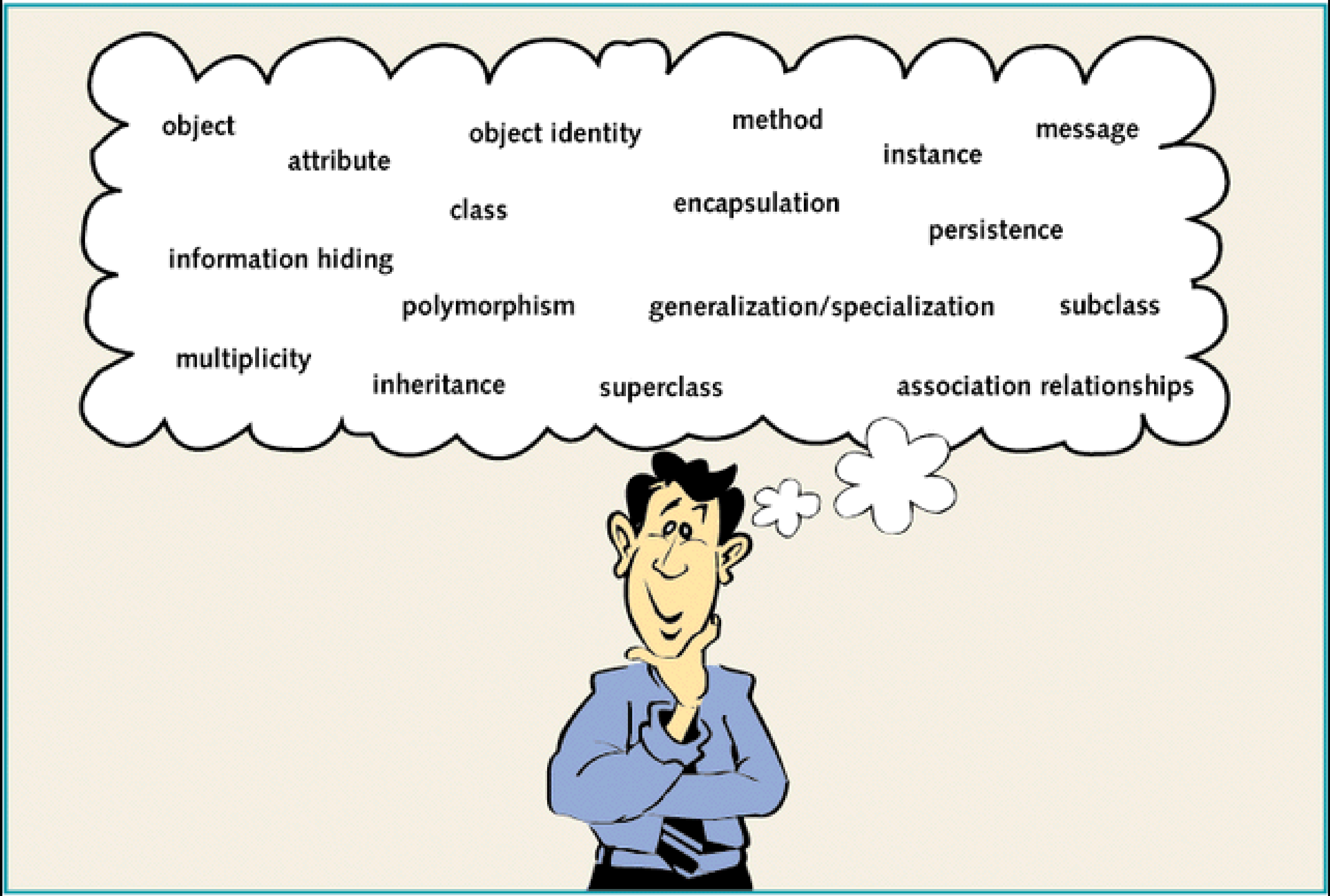


Figure 1-5 Key OO concepts

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
  - objects that make up the user interface
  - e.g. buttons, labels, windows, etc.

- Problem Domain objects
  - Objects that represent a business application
  - 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**

What is Object-Orientation?

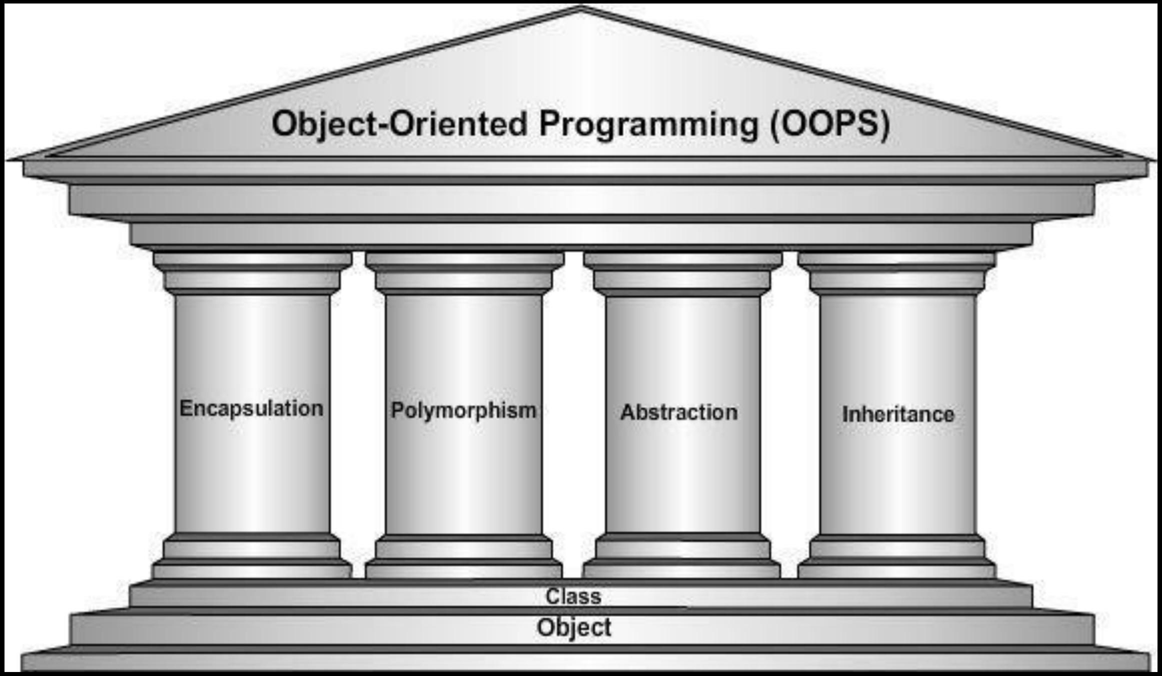
- What is Object?

- An "object" is anything to which a concept applies, ***in our awareness***
- Things drawn from the problem domain or solution space.
  - E.g., a living person in the problem domain, a software component in the solution space.

- A structure that has identity and properties and behavior
- It is an instance of a collective concept, i.e., a class.

OO Principles

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

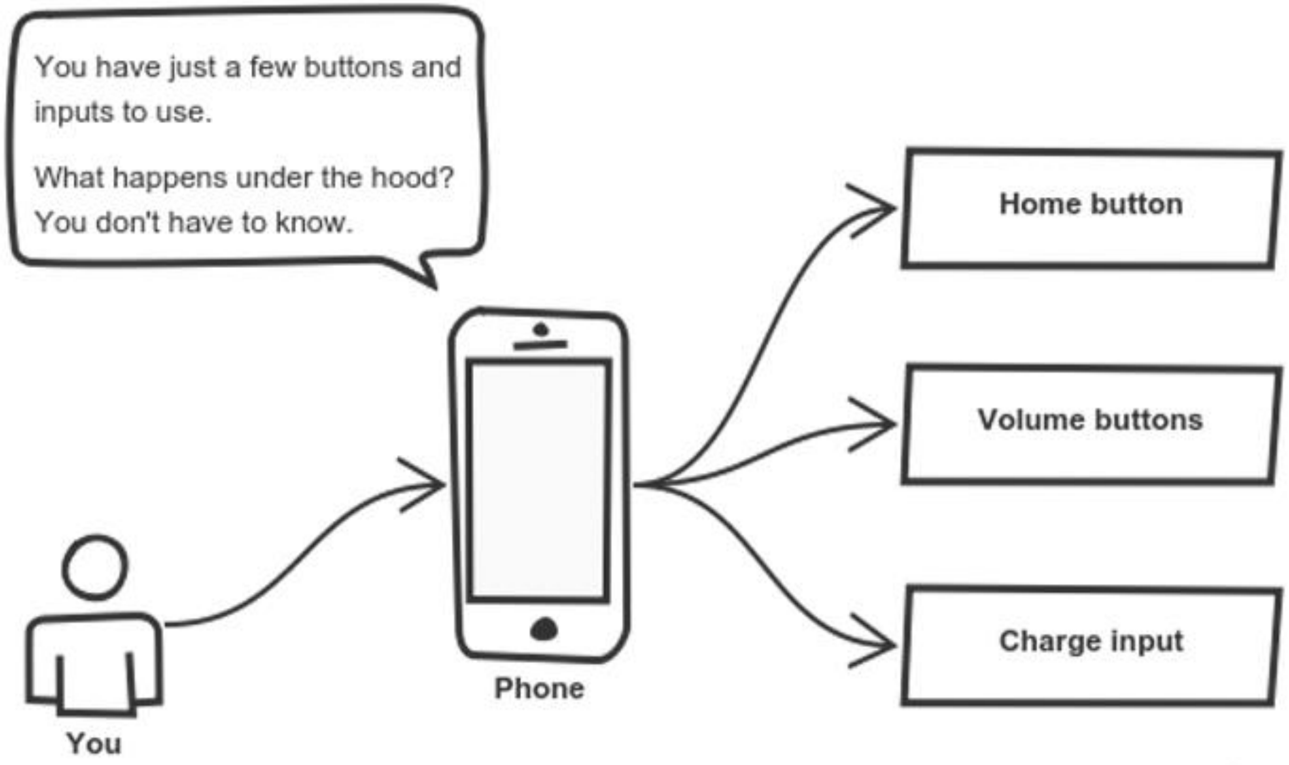


Principle 1: Abstraction

- 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.

Difference between Abstraction and Encapsulation

Abstraction	Encapsulation
1. Abstraction solves the problem in the design level.	1. Encapsulation solves the problem in the implementation level.
2. Abstraction is used for hiding the unwanted data and giving relevant data.	2. Encapsulation means hiding the code and data into a single unit to protect the data from outside world.
3. Abstraction lets you focus on what the object does instead of how it does it	3. Encapsulation means hiding the internal details or mechanics of how an object does something.
4. **Abstraction**- Outer layout, used in terms of design. For Example:- Outer Look of a Mobile Phone, like it has a display screen and keypad buttons to dial a number.	4. **Encapsulation**- Inner layout, used in terms of implementation. For Example:- Inner Implementation detail of a Mobile Phone, how keypad button and Display Screen are connect with each other using circuits.

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!!**

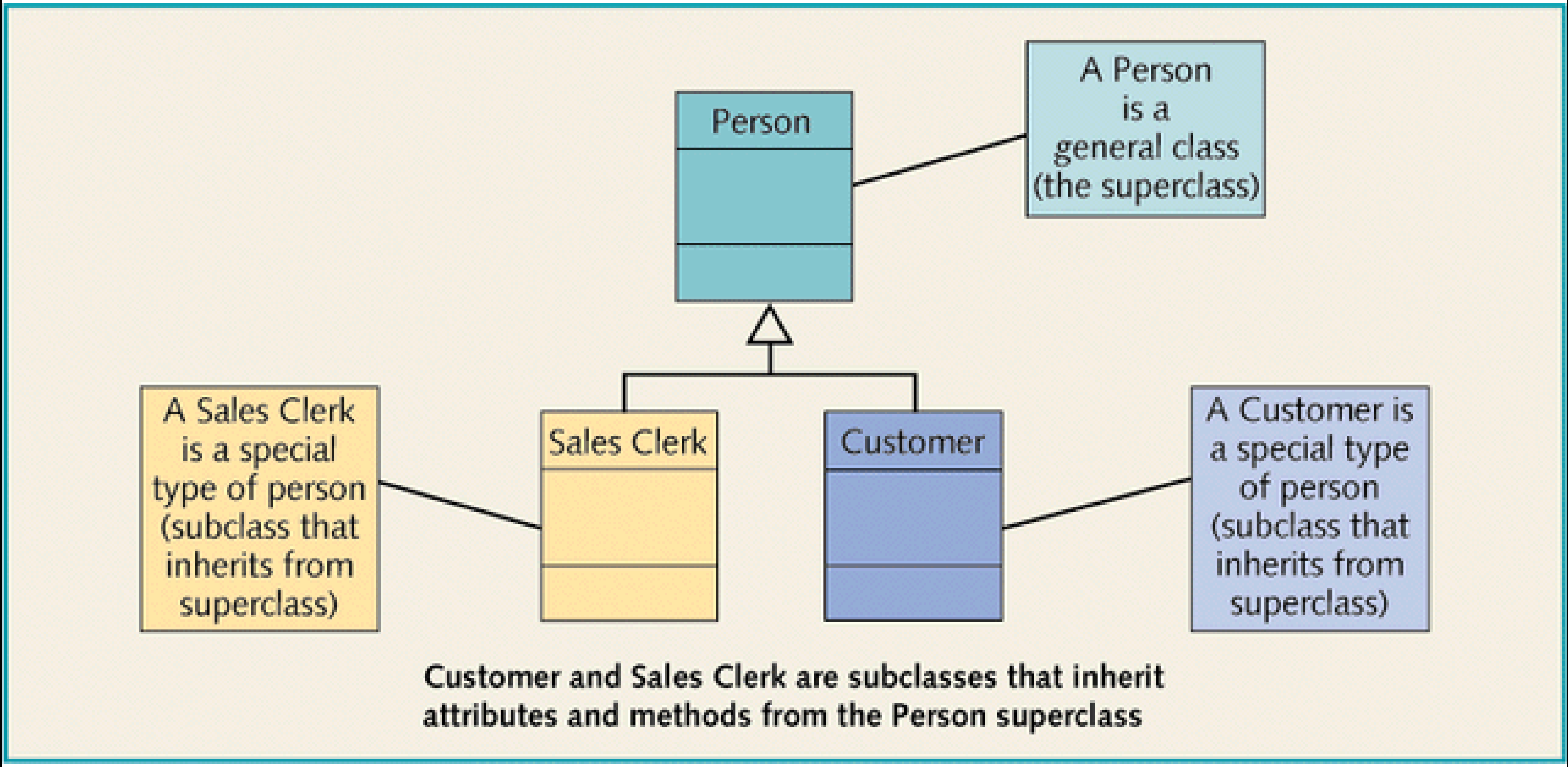

Person
A Person is a general class (the superclass)
A Sales Clerk is a special type of person (subclass that inherits from superclass)
Sales Clerk
Customer
A Customer is a special type of person (subclass that inherits from superclass)
Customer and Sales Clerk are subclasses that inherit attributes and methods from the Person superclass

Figure 1-11 Superclass and subclass

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

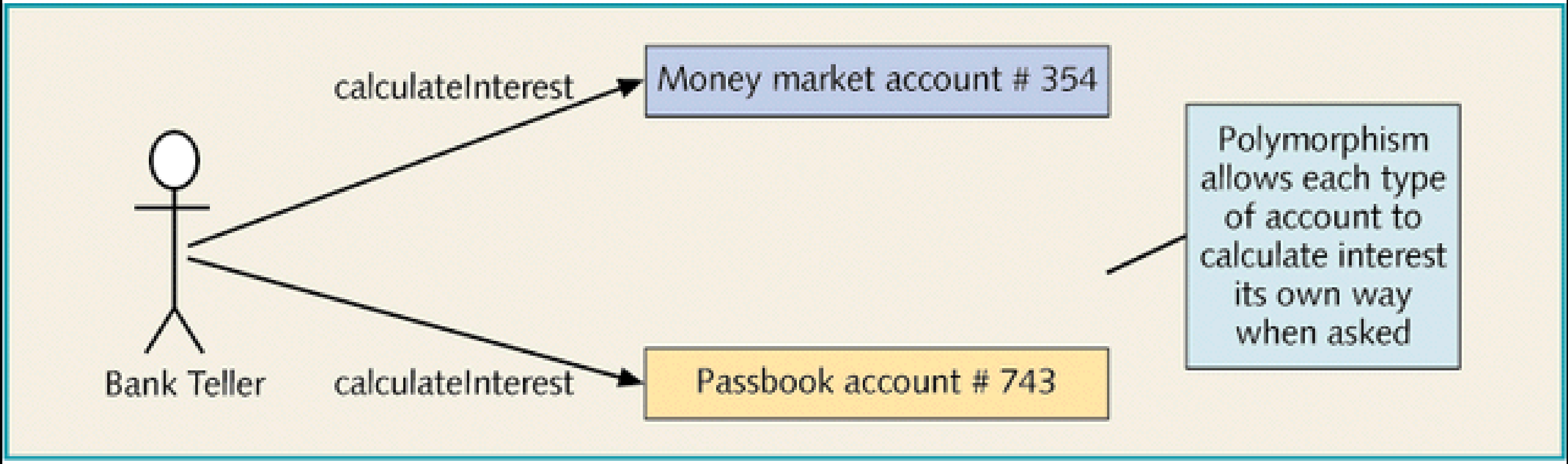
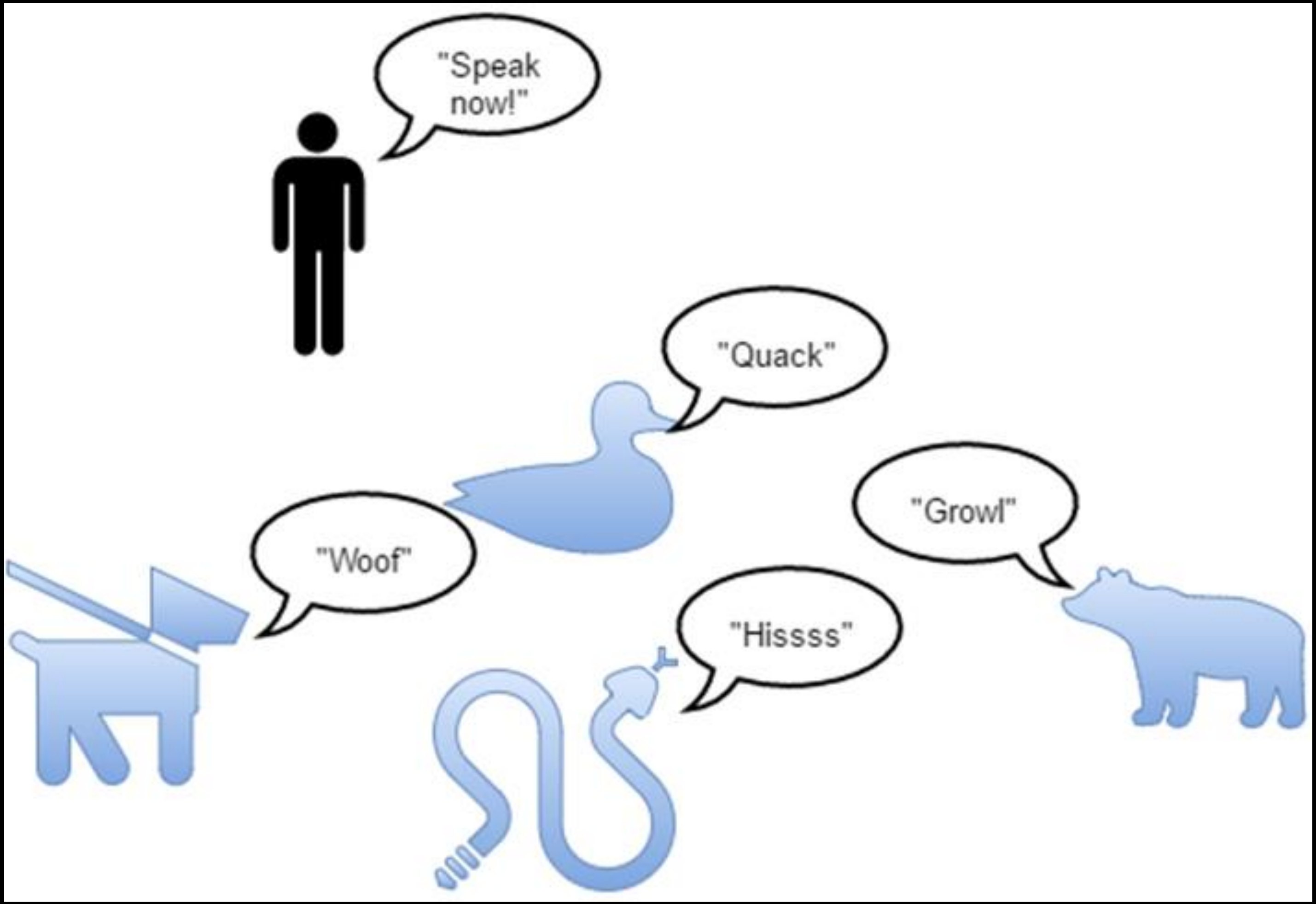

calculateInterest
Money market account # 354
Polymorphism allows each type of account to calculate interest its own way when asked
Bank Teller
calculateInterest
Passbook account # 743

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

Polymorphism

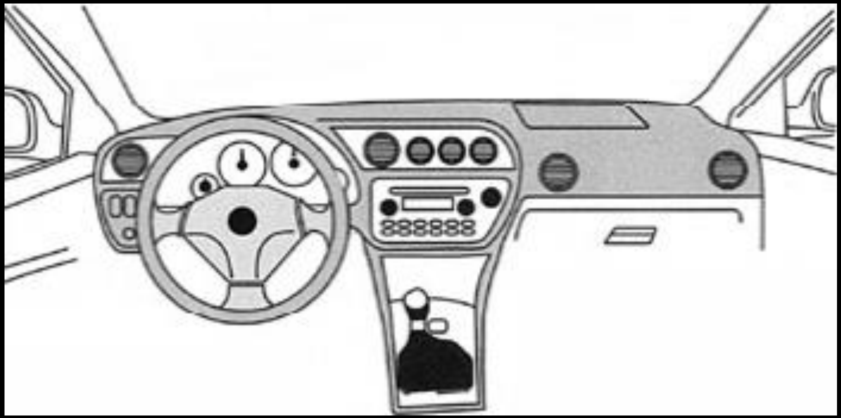
- Objects of different classes respond to the same message differently.
- ability to dynamically choose the method for an operation at run-time or service-time



What is Object-Orientation

- Interfaces

- Information hiding - all data should be hidden within a class,
- make all data attributes private
- provide public methods to get and set the data values
  - e.g. Grade information is usually confidential, hence it should be kept private to the student. Access to the grade information should be done through *interfaces*, such as setGrade and getGrade



That is all