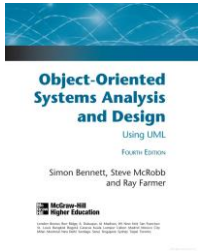
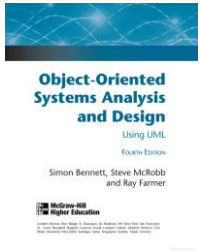


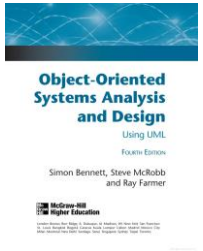
Lecture 2- Object Orientation

Topic covered



- ✧ Object-Orientation Concepts
- ✧ Object-Orientation Benefits

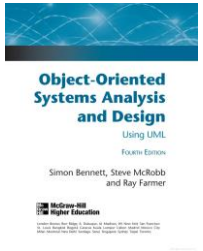
Object



- An object is “an abstraction of something in a problem domain, reflecting the capabilities of the system to keep information about it, interact with it, or both.”

- “We define an object as a concept, abstraction, or thing with crisp boundaries and meaning for the problem at hand. Objects serve two purposes: they promote understanding of the real world and provide a practical basis for computer implementation.”

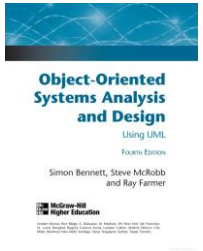
Object



✧ “Objects have state, behavior and identity.”

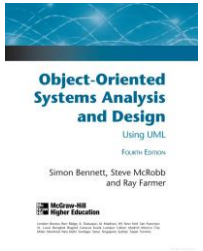
- Identity (**Who am I?**)
  - each object is unique
- State (**What do I know?**)
  - the conditions of an object at any moment that affect how it behaves
- Behaviour (**What can I do?**)
  - the way in which an object responds to messages

Object



Object	Identity	States	Behaviour
A person	'Hussain Pervez'	Studying Resting Qualified	Speak Walk Read
A shirt	'My favourite button-down white denim shirt'	Pressed Dirty Worn	Shrink Stain Rip
A sale	'Sale no 0015, 15/06/02'	Invoiced Cancelled	Earn loyalty points
A bottle of ketchup	'*This* bottle of ketchup'	Unsold Opened Empty	Spill in transit

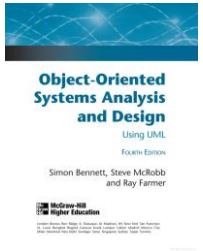
Class



- A class is “a set of objects that share the same specifications of features (attributes, operations, links), constraints (e.g. when and whether an object can be instantiated)”

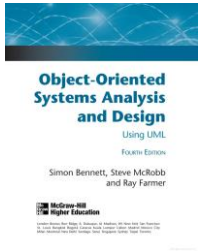
- Moreover, “The purpose of a class is to specify a classification of objects an to specify the features that characterize the structure and behavior of those objects”

Class



- An object = An instance of some class
  - Every object must be an instance of some class
- A class = A set of objects that share the same
  - structure
    - what information it holds
    - what links it has to other objects
  - behaviour
    - what things it can do

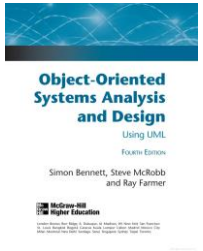
Message Passing



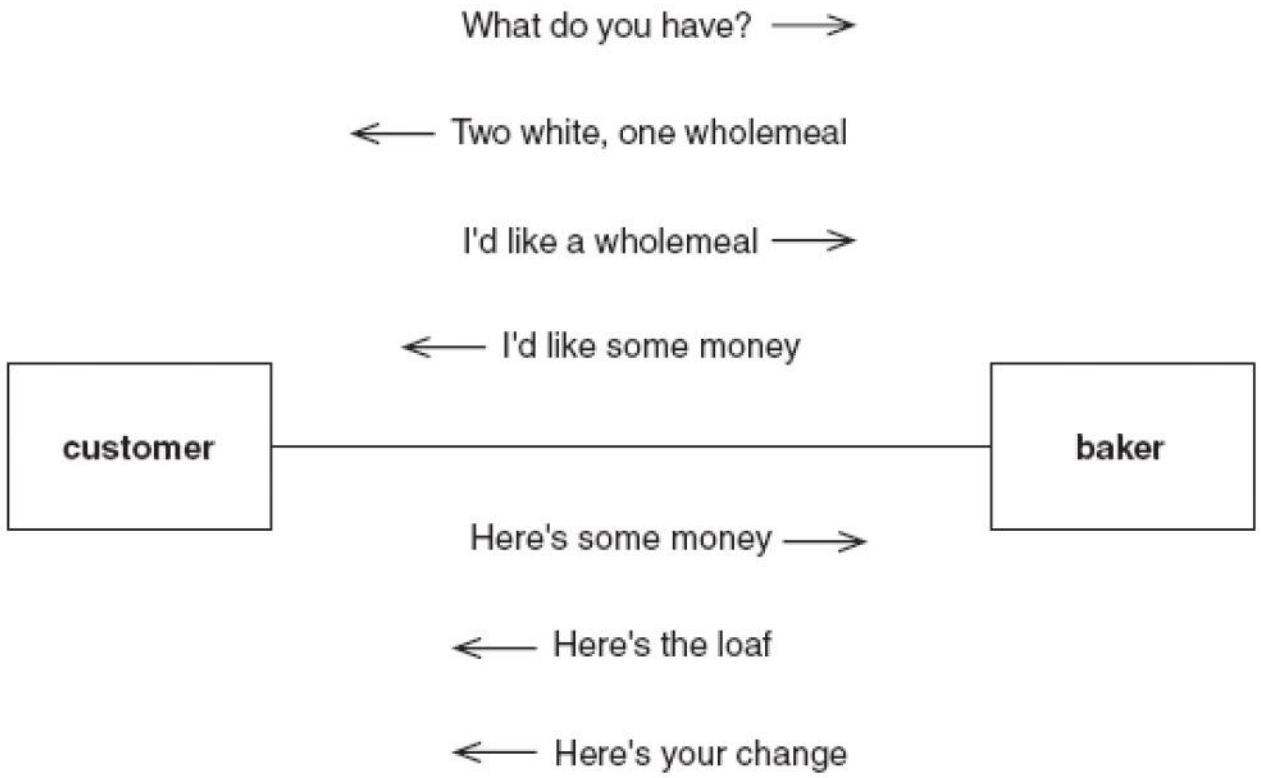
✧ Objects collaborate to fulfil some system function, and they communicate by sending each other messages:

- A *question* message asks an object for some information
  - How much is the balance?
- A *command* message tells an object to do something
  - Withdraw 100 pounds

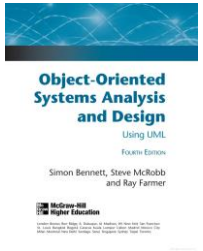
Message Passing: Example



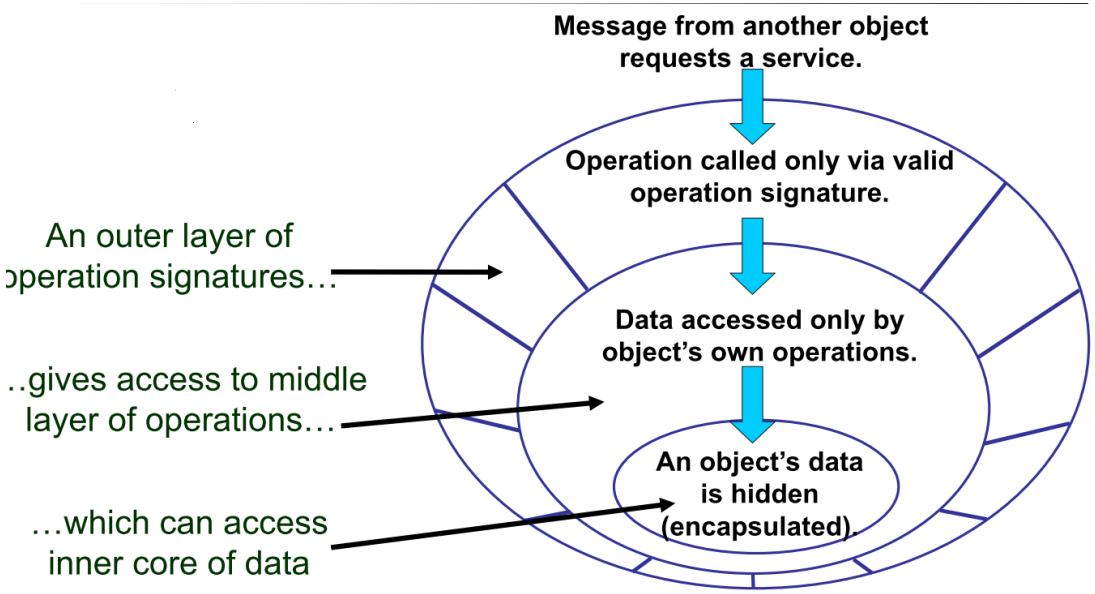
- Buying a loaf of bread:



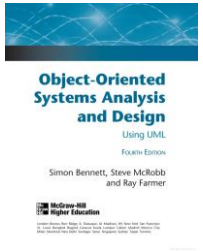
Encapsulation



✧ ‘Layers of an onion’ model of an object:

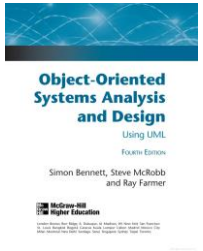


Inheritance



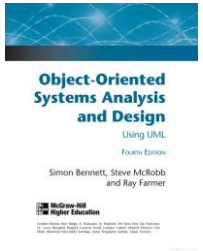
- A subclass always inherits all the characteristics (data structure and behavior) of all its superclasses.
- The definition of a subclass should always include at least one detail not derived from any of its superclasses.

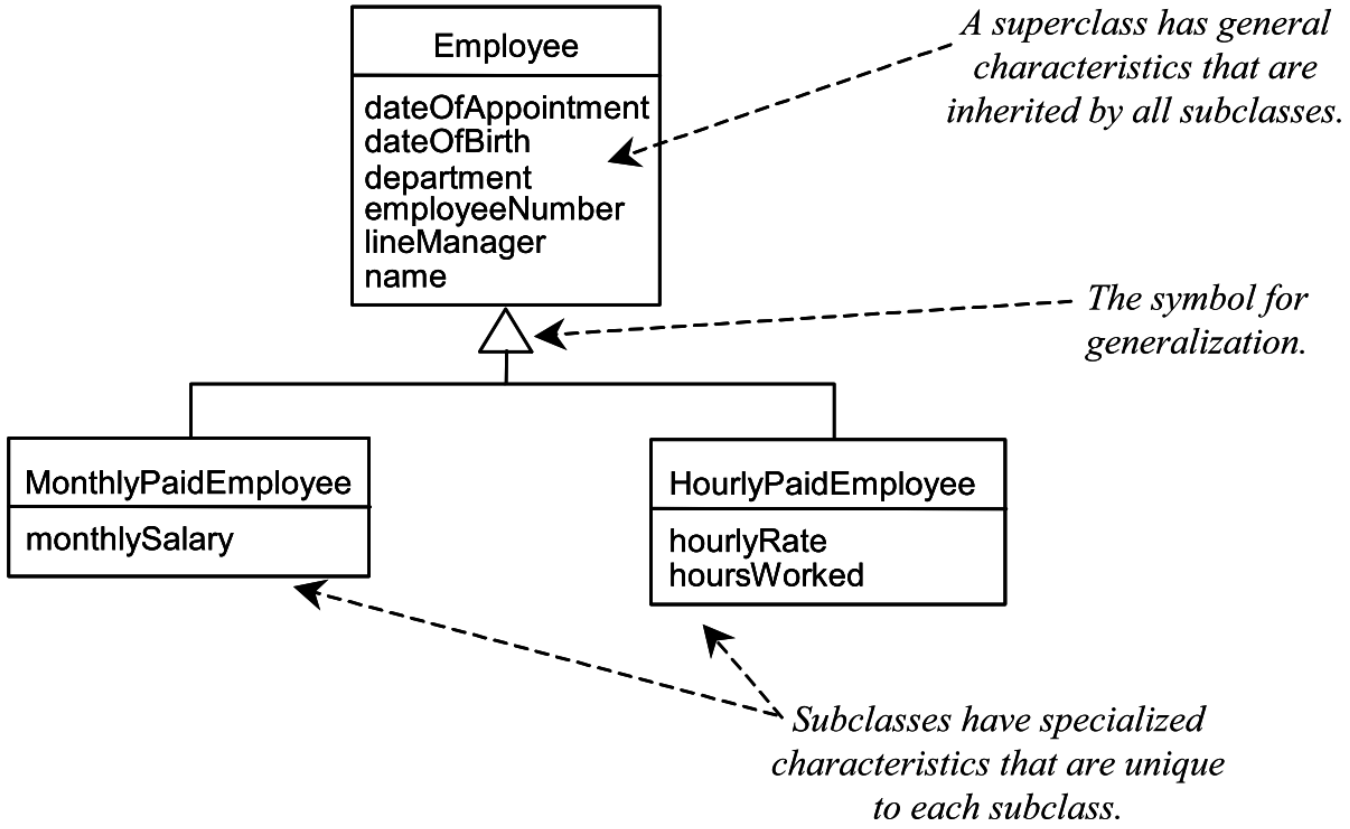
Generalization



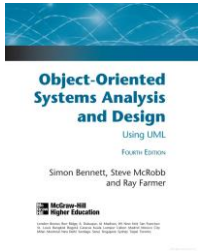
✧ A subclass inherits the structure and behavior of its superclass.

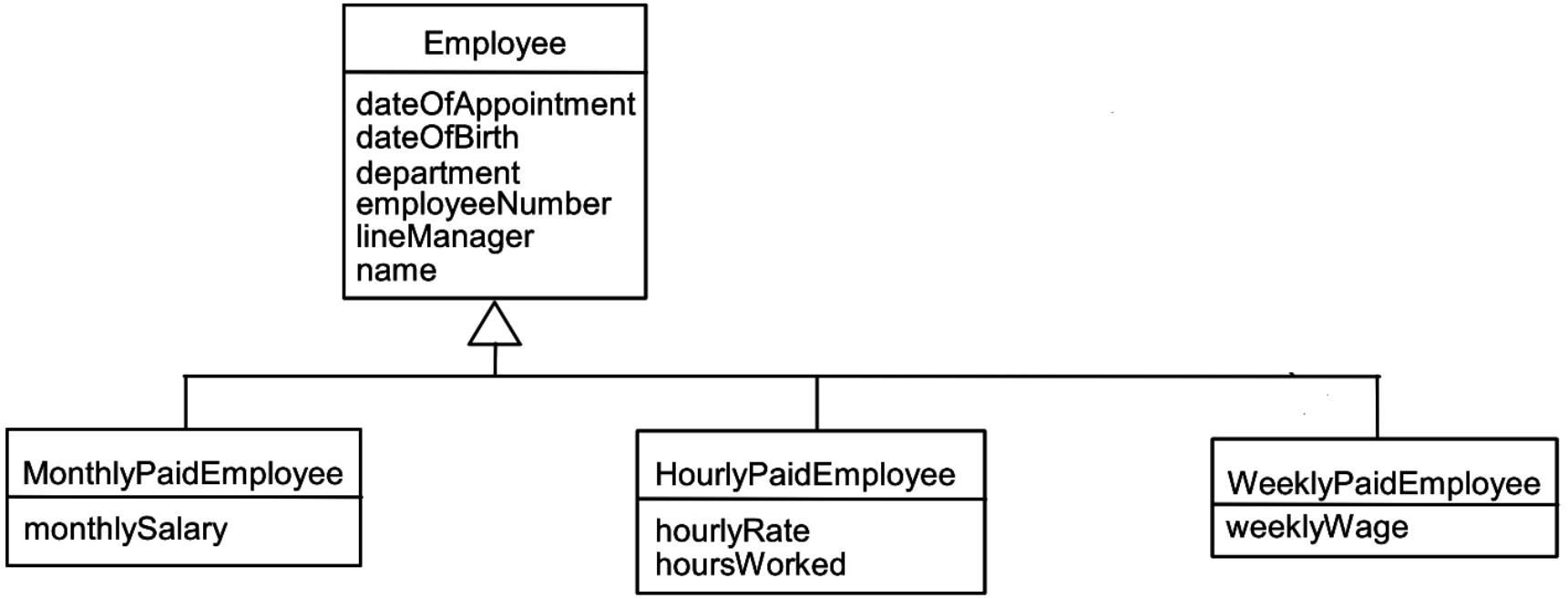
Generalization in UML



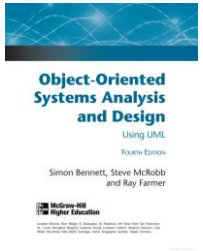


Advantages of using Generalization



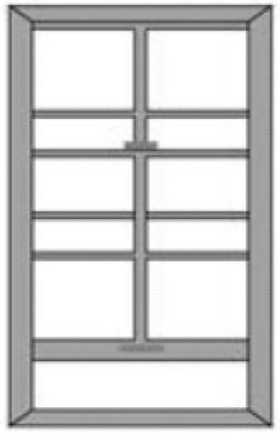


Polymorphism



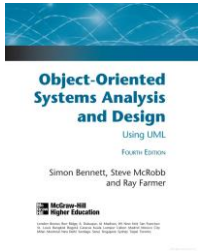
- ✧ Polymorphism allows one message to be sent to objects of different classes
- ✧ Each receiving object responds appropriately, i.e., different kinds of objects may respond to the message in different ways.

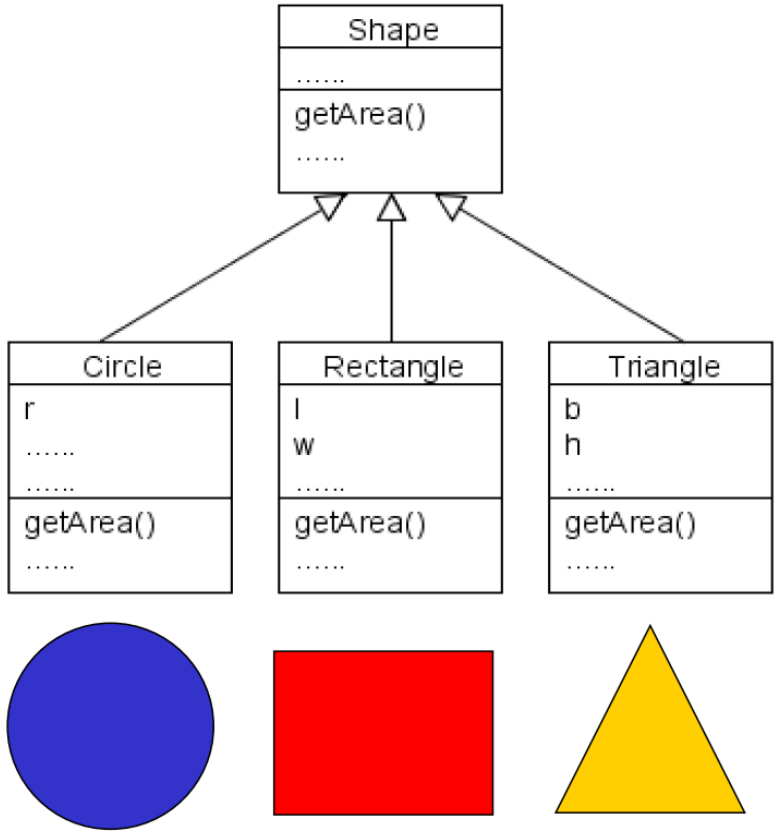
Polymorphism: Example

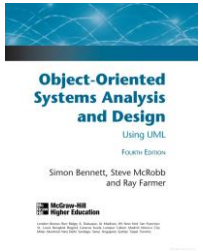


open

Polymorphism: Example

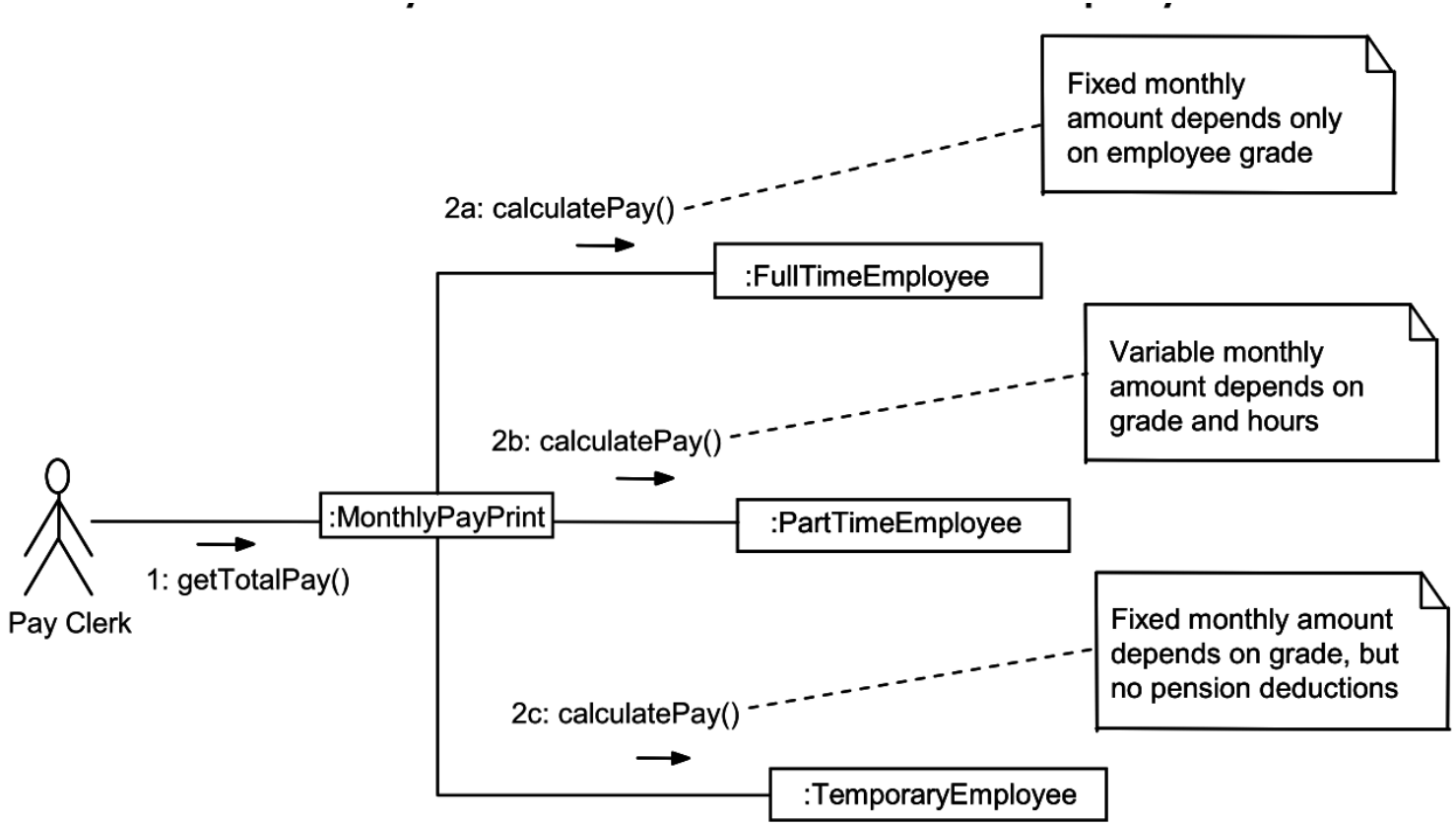




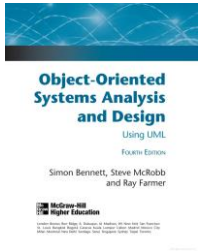


Polymorphism: Example

✧ calculatePay" for different kinds of employees



Benefits of Object-Orientation



- ✧ Object-Orientation concepts and techniques improve both software quality and software productivity.
- ✧ Modularity for easier troubleshooting
- ✧ Reuse of code through inheritance
- ✧ Flexibility through polymorphism
- ✧ Effective problem solving