Unified Modelling Language (UML)

COMP 1531, 17s2

Aarthi Natarajan

Week 6 Wednesday

UML stands for **Unified Modelling Language**

Programming languages not abstract enough for OO design

An open source, graphical language to model software solutions, application structures, system behaviour and business processes

Several uses:

- As a design that communicates aspects of your system
- As a software blue print
- Sometimes, used for auto code-generation

Agile Modelling - Core Practices

Modelling is vital for software development – a picture is worth a thousand words

Core practices to bear in mind during AM

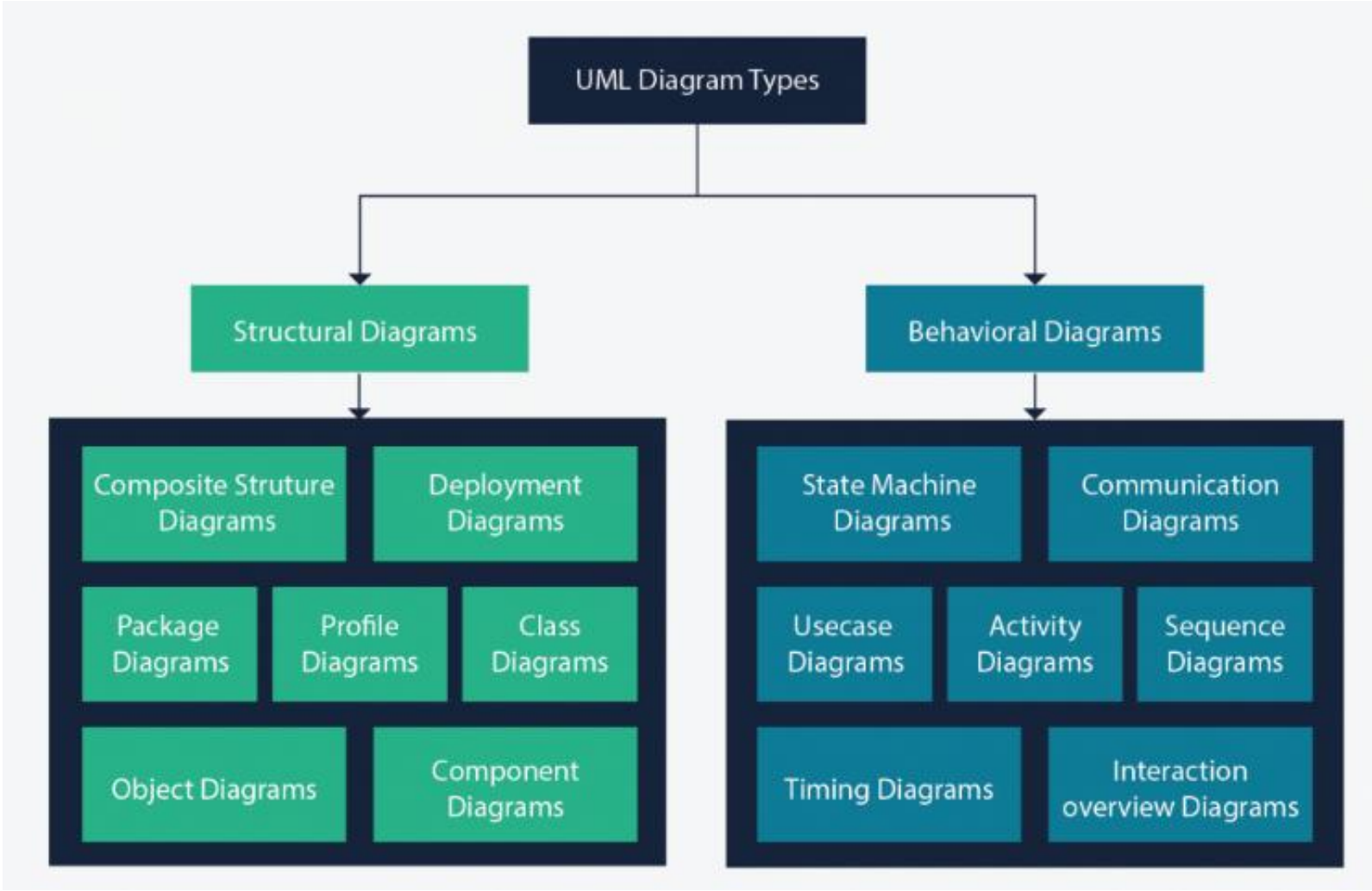
- Apply the right artifact
- Iterate to another model
- Use the simplest tools
- Model in small increments
- Create several models in parallel
- Depict models simply

UML diagrams can be broadly classified as:

- **Structure Diagrams**
  - diagrams that depict the elements of a specification that are irrespective of time (e.g. class diagram)

- **Behaviour Diagrams**
  - diagram that depicts behavioural features of a system or business process (e.g. use case diagram or class diagram)
  - a subset of these diagrams are referred to as interaction diagrams that emphasis interaction between objects

UML Diagram Types



Class Diagrams

A UML class diagram is useful for showing

- the classes in the system
- their attributes and methods
- relationships between classes

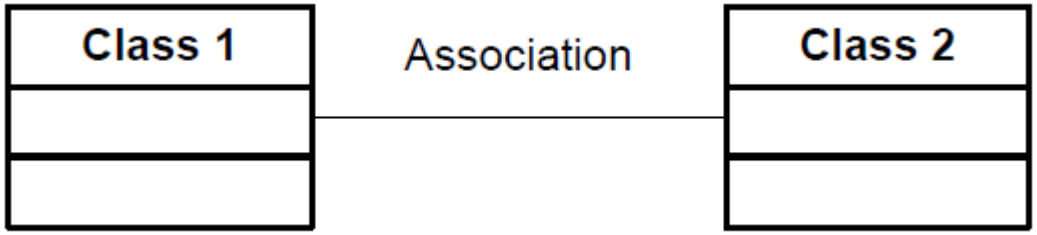
A UML class diagram does not include:

- Details of interactions between classes or how a particular behaviour is implemented

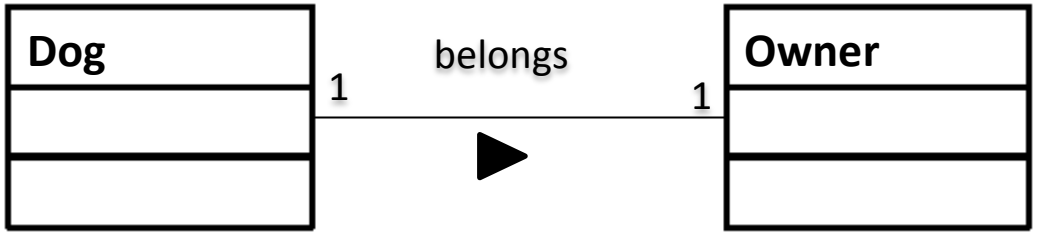
Simple steps to create a Class Diagram

- Use CRC technique to identify the classes, their attributes and responsibilities and collaborations
- Draw a conceptual class diagram
  - Put classes in rectangles and draw associations (collaborations) between classes
  - Fill in multiplicity (Number of occurrences of one class for a single occurrence of the associated class
  - Identify attributes
  - Identify methods (responsibilities)

Elements of a class diagram

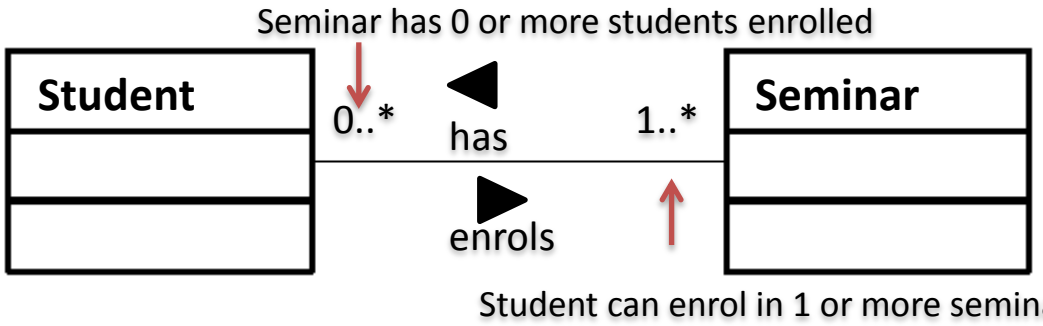


- A relationship between two classes represented as a line between the classes



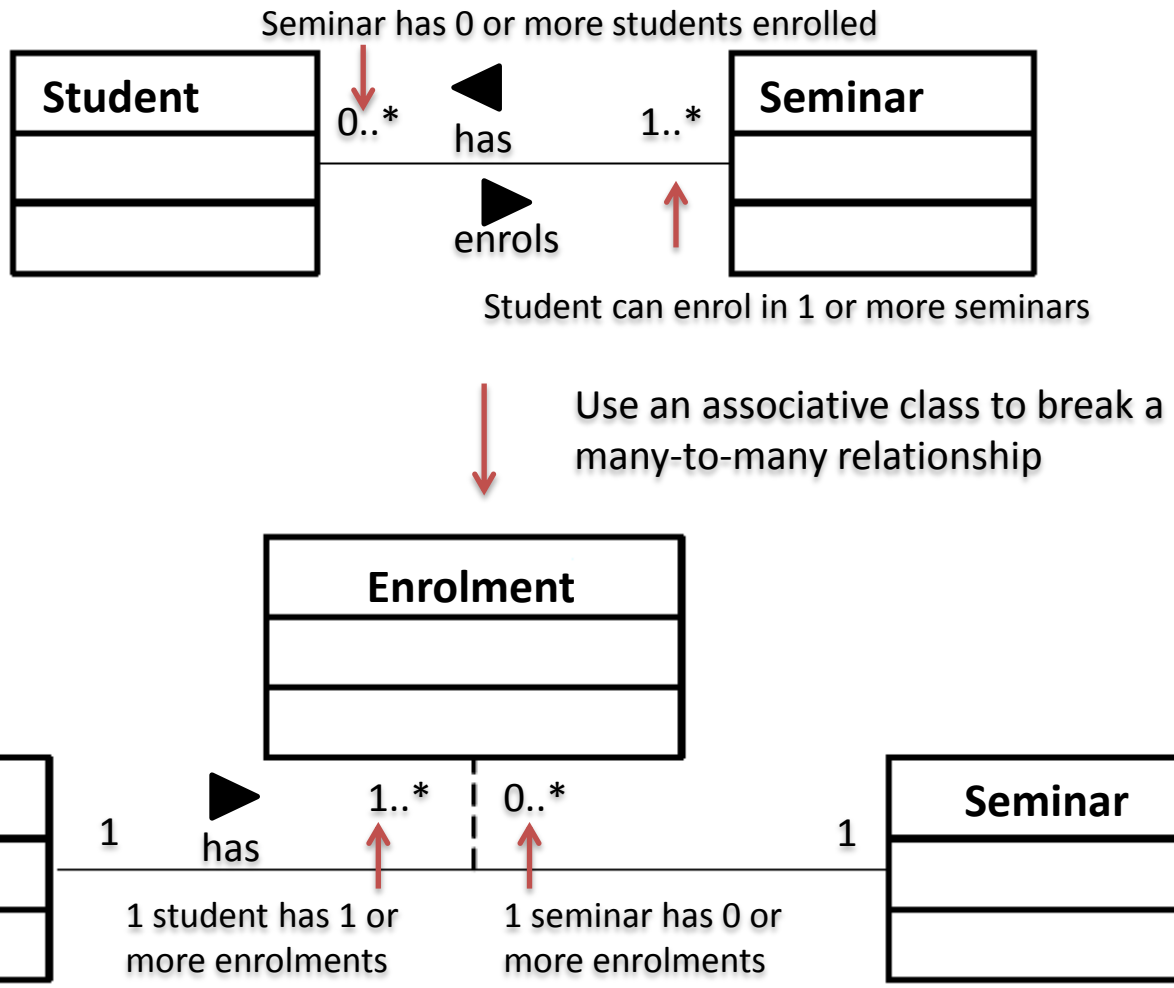
- A one-to-one association between 2 classes

- A one-to-many association between 2 classes

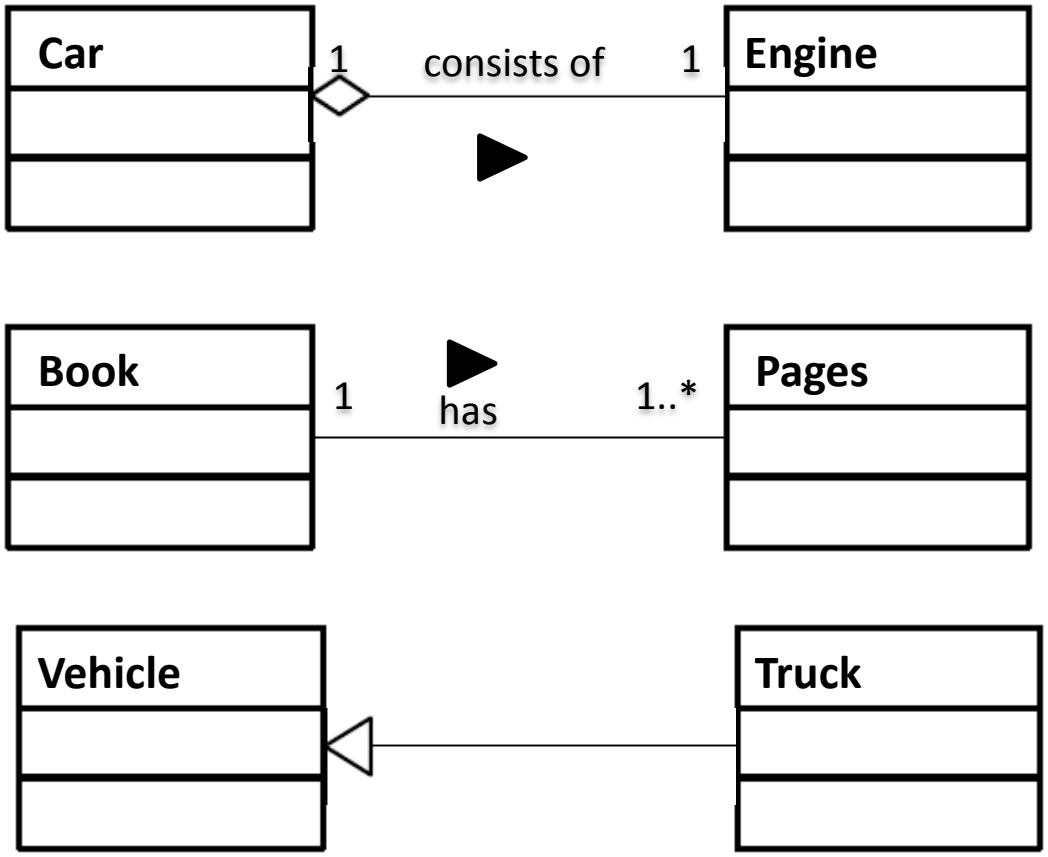


- A bidirectional many-to-many association between 2 classes

Elements of a class diagram

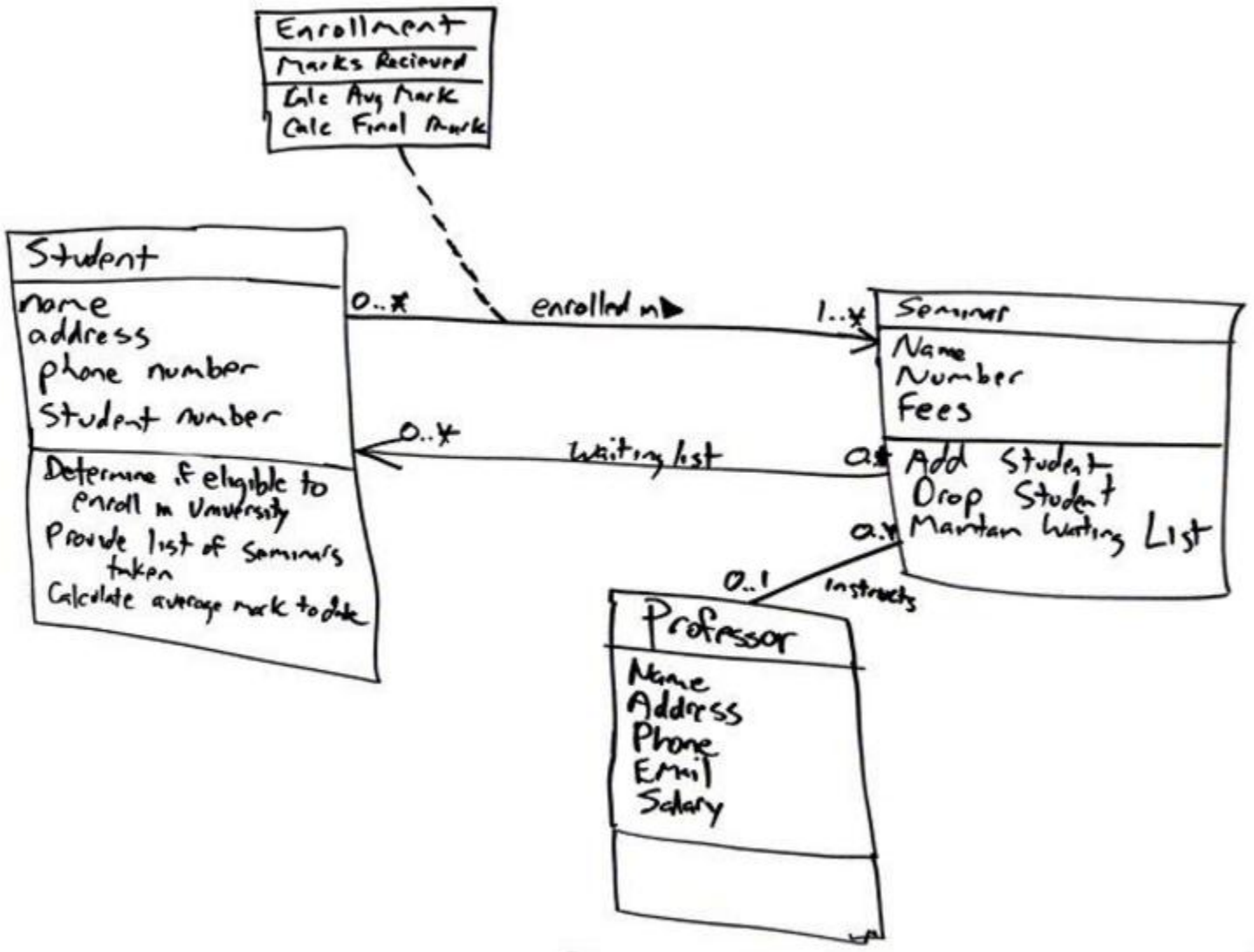


Elements of a class diagram



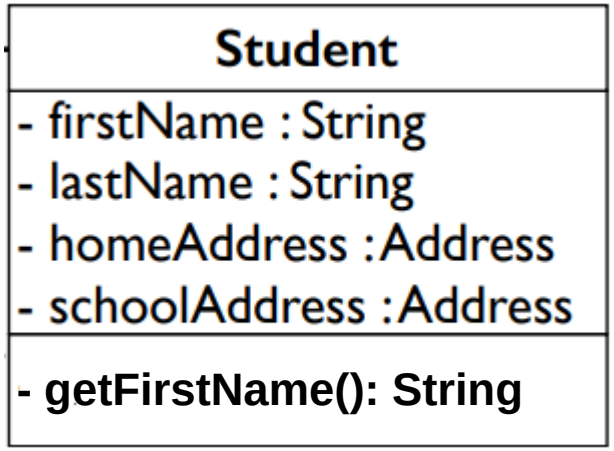
- Aggregation - “has-a” relationship where the part can exist without container
- Composition – “is-composed-of” relationship where part cannot live without container
- Generalization (Inheritance) – “is-a-kind-of” relationship

Conceptual Class Diagram – remember model simply and apply the right artifact practices



Detailing the classes for a design class diagram

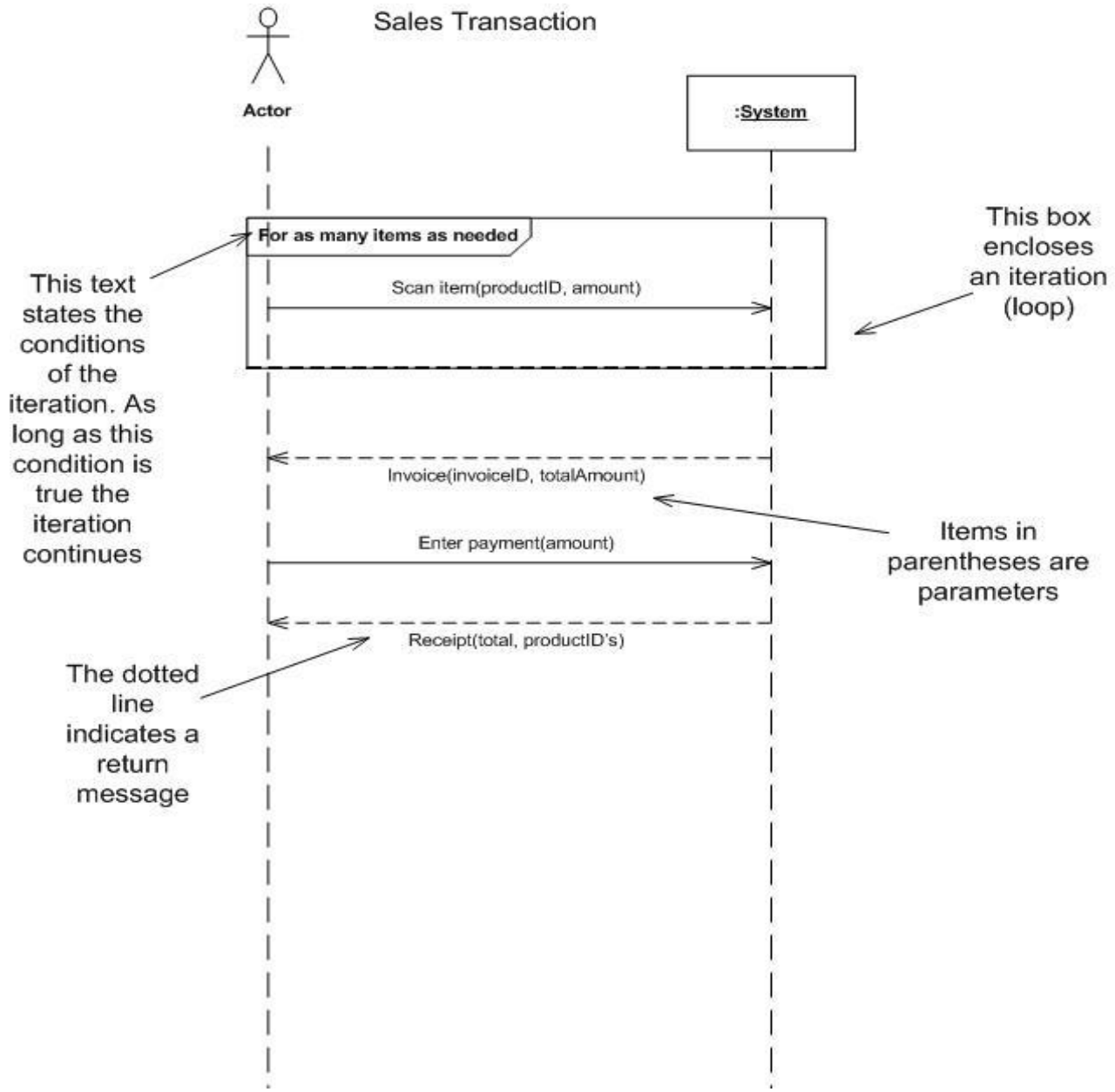
- A conceptual class diagram is evolved into a detailed design class diagram
- Here, the attributes and methods are specified clearly along with **visibility** (through use of access modifiers e.g. private(-), public (+))
- Syntax for specifying an attribute:
  - <visibility> attribute_name : data_type
- Syntax for specifying a method:
  - <visibility> method_name(parameters): return_type



Sequence Diagram

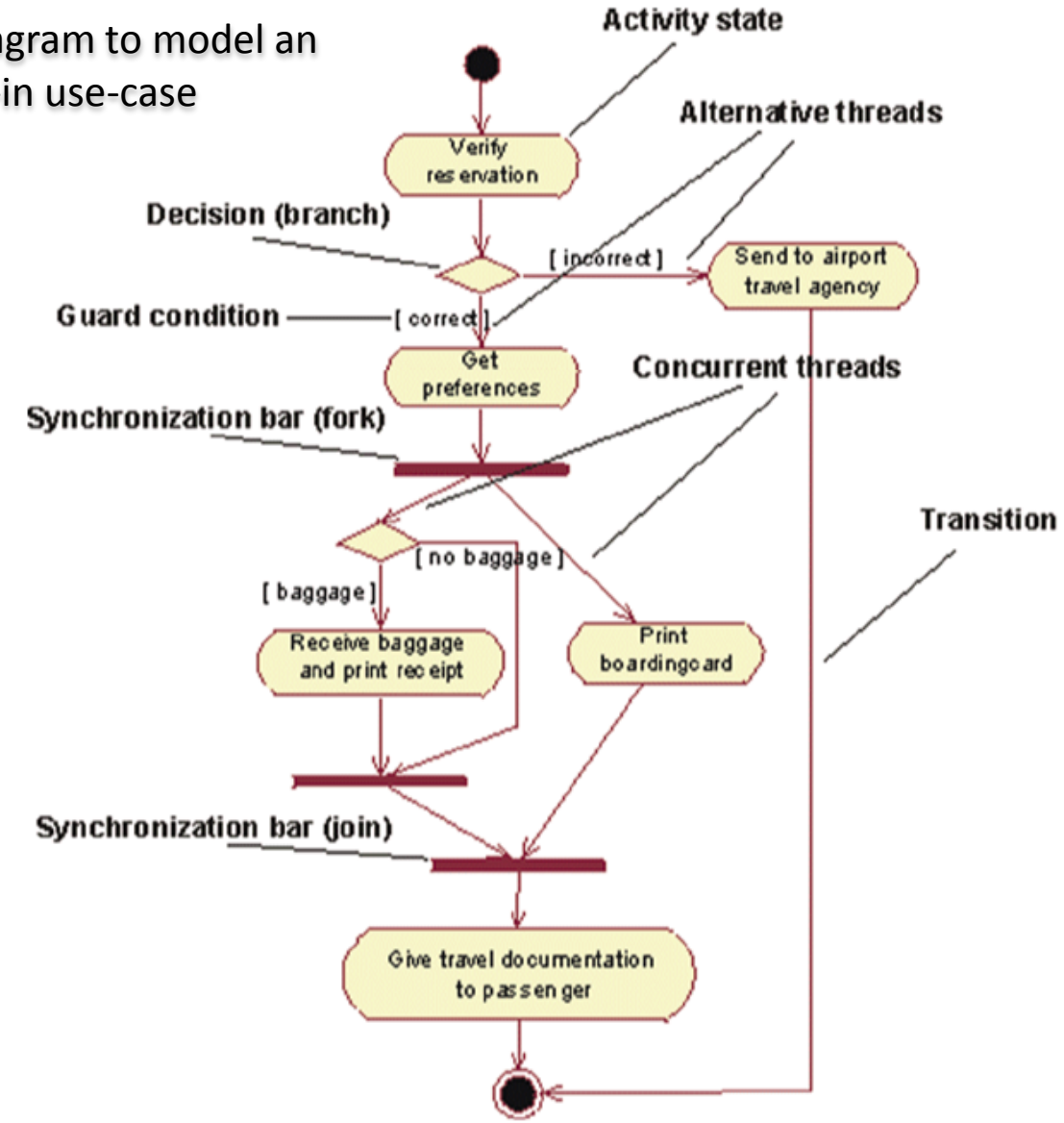
- Is a UML behaviour diagram and provides a visual summary of a use-case scenario
- Shows for a particular use case scenario the events the external actors generate, their order and possible inter-system events
- All systems are treated as a black box; the diagram places emphasis on events that cross the system boundary from actors to systems.
- Should be done for the main success scenario of the use case, and frequent or complex alternative scenarios.
- A system sequence diagram should specify and show the following:
  - External actors
  - Messages (methods) invoked by these actors
  - Return values (if any) associated with previous messages
  - Indication of any loops or iteration area

Example of a sequence diagram



Activity Diagrams are useful to visualise the workflow of a business use-case

An activity diagram to model an airport check-in use-case



Activity Diagram vs Sequence Diagram

- Both can be perceived for a similar purpose and can be viewed as complementary techniques
- Activity diagrams give focus to the workflow while sequence diagrams give focus to handling of business entities
- an activity diagram with partitions focuses on how you divide responsibilities onto classes, while the sequence diagram helps you understand how objects interact and in what sequence.