Introduction to Object Oriented Analysis

- Object oriented languages (Java & .NET) are growing in popularity.
  - Because object oriented programming
    - can promote better code reuse to hold down program costs
    - More appropriate for projects where geographically separated groups of programmers have to collaborate to produce an integrated system and each team can develop pieces of programming code independently

Introduction to Object Oriented Analysis

- Object oriented approach to programming requires techniques for Object oriented Analysis and Design
- OOA is an approach used to
  - Study existing objects to see if they can be reused or adapted for new users &
  - Define new / modified objects that will be combined with existing objects into a useful business computing application

Introduction to Object Oriented Analysis

- Some of the object oriented diagrams, such as class and sequence diagrams would be inappropriate except when the system will be implemented in an OO environment
- Other diagrams developed for OOAD can be used in any kind of environment
  - e.g. use cases

Introduction to Unified Modeling Language (UML)

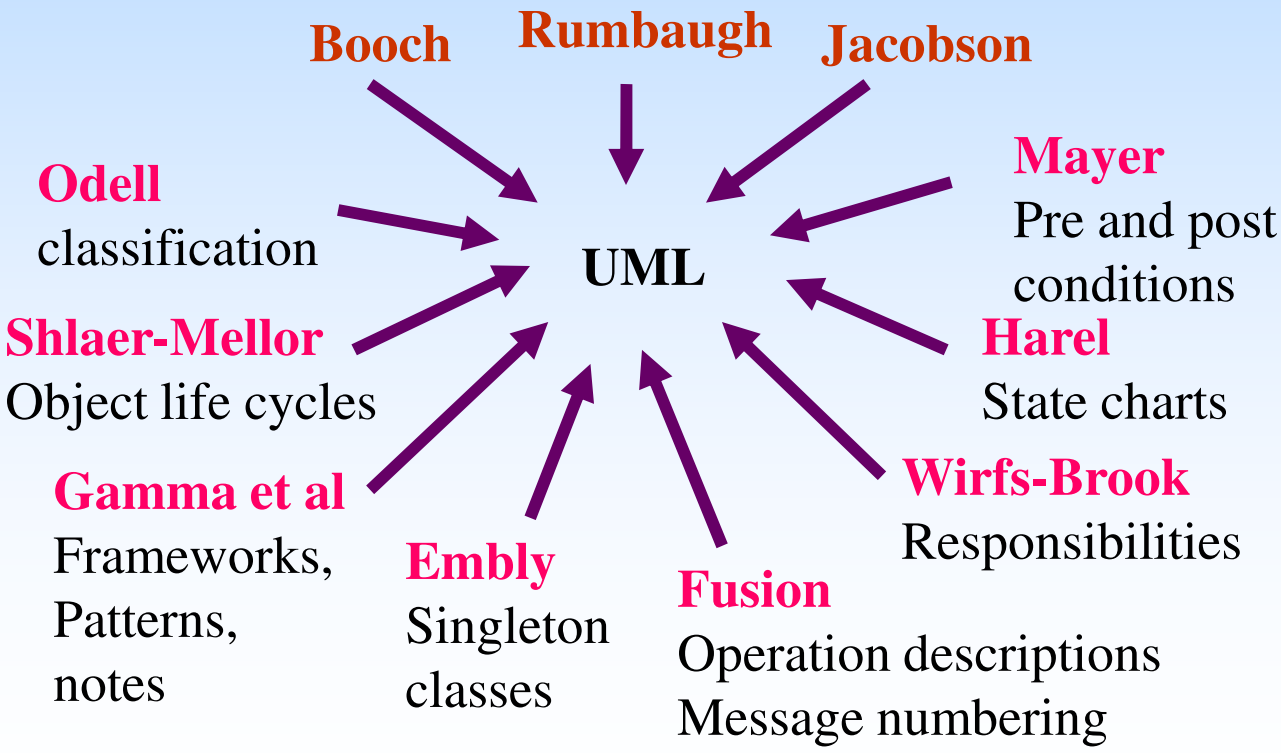
- One of the most exciting and useful tools in the world of system development
- A visual modeling language
- Enables system builders to create blueprints that capture their visions in a standard, easy-to-understand way
- Provides a mechanism to effectively share and communicate these visions with others

Introduction to Unified Modeling Language (UML)

- It most directly unifies the methods of
  - Booch,
  - Rumbaugh (OMT) and
  - Jacobson

  as well as the best ideas from a number of other methodologies.

UML Inputs


Booch
Rumbaugh
Jacobson
Odell
classification
Mayer
Pre and post
conditions
UML
Shlaer-Mellor
Object life cycles
Harel
State charts
Gamma et al
Frameworks,
Patterns,
notes
Embly
Singleton
classes
Fusion
Operation descriptions
Message numbering
Wirfs-Brook
Responsibilities

UML

- **The UML is an attempt to standardize the artifacts of analysis and design :**

  ie. Semantic models,

  Syntactic notations,

  and diagrams

UML History

- The first public draft version – (version 0.8) – Oct 1995
- Feedback from public and Ivor Jacobson's inputs included – (ver. 0.9 Jul 1996, ver. 0.91 Oct 1996)
- Ver 1.0/1.1 was presented to OMG group for standardization in July/Sep 1997.
- Nov 1997 – UML adopted as the standard modelling language by OMG
- Ver 1.2 - June 1998
- Ver 1.3 – Dec 1998
- Ver 1.4 – 2000
- Ver 2.0 – 2003
- Ver 2.1 – 2007
- Ver 2.2 - 2009
- Ver 2.3 – 2010,
- Ver 2.4 -2011 Jan
- UML 2.5 -2012 October

UCSC

UML
Components and Capabilities

- The UML consists of a number of graphical elements that combine to form diagrams.
- Because it is a language, the UML has rules for combining these elements.
- Lets look at the diagrams before looking at the elements and rules. You will be using these diagrams to do systems analysis.
- The purpose of the diagrams is to present multiple views (model) of a system.
- UML 2.* include many different diagrams.

UML Diagrams

- UML has 14 types of diagrams divided into two categories.

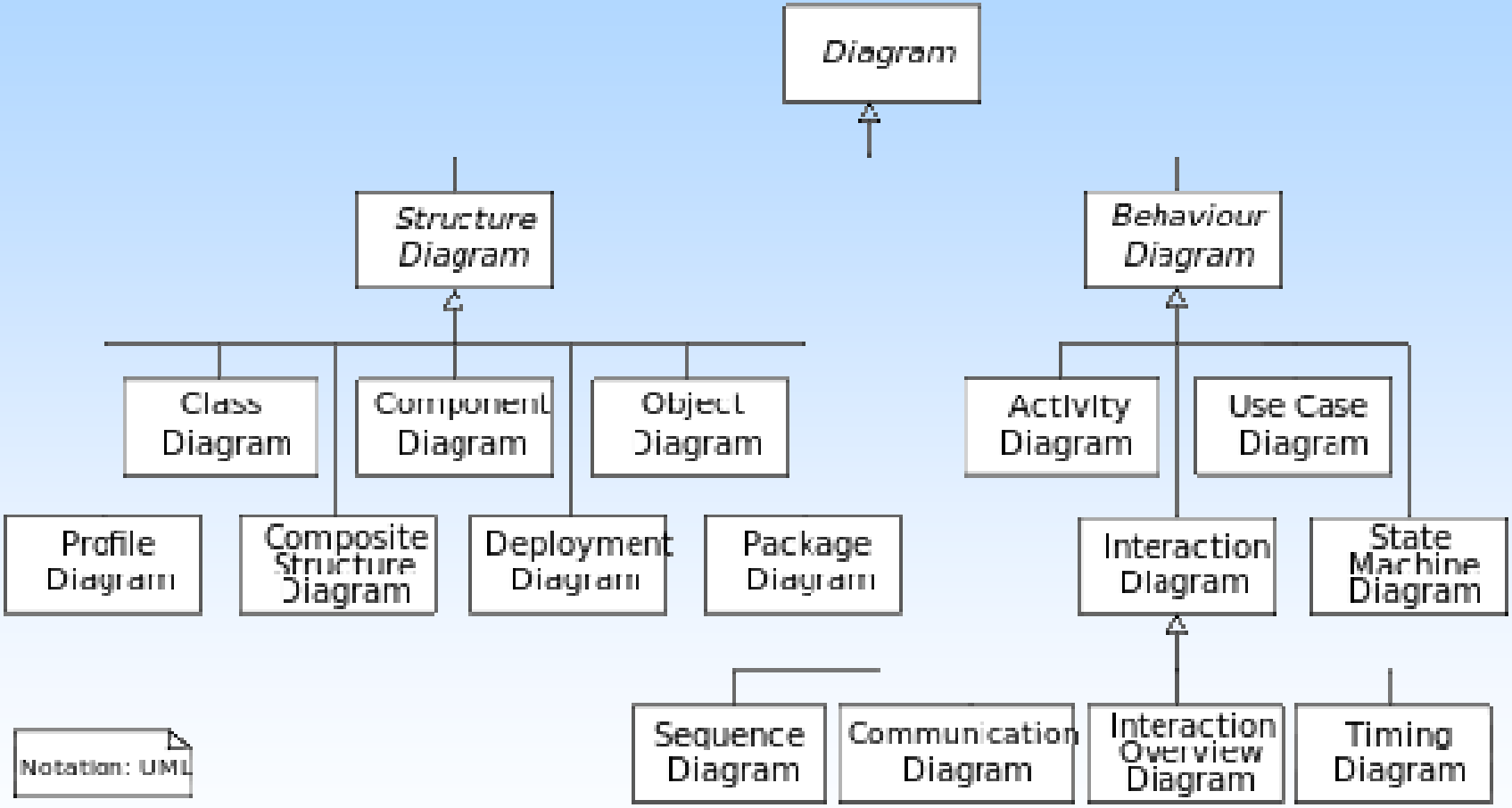
  - Structure Diagrams :

  Seven diagram types represent *structural* information,

  -Behaviour Diagrams :

  other seven represent general types of *behavior*..

UML Diagrams



UML Diagrams

- UML include many different diagrams.

  Class Diagrams :

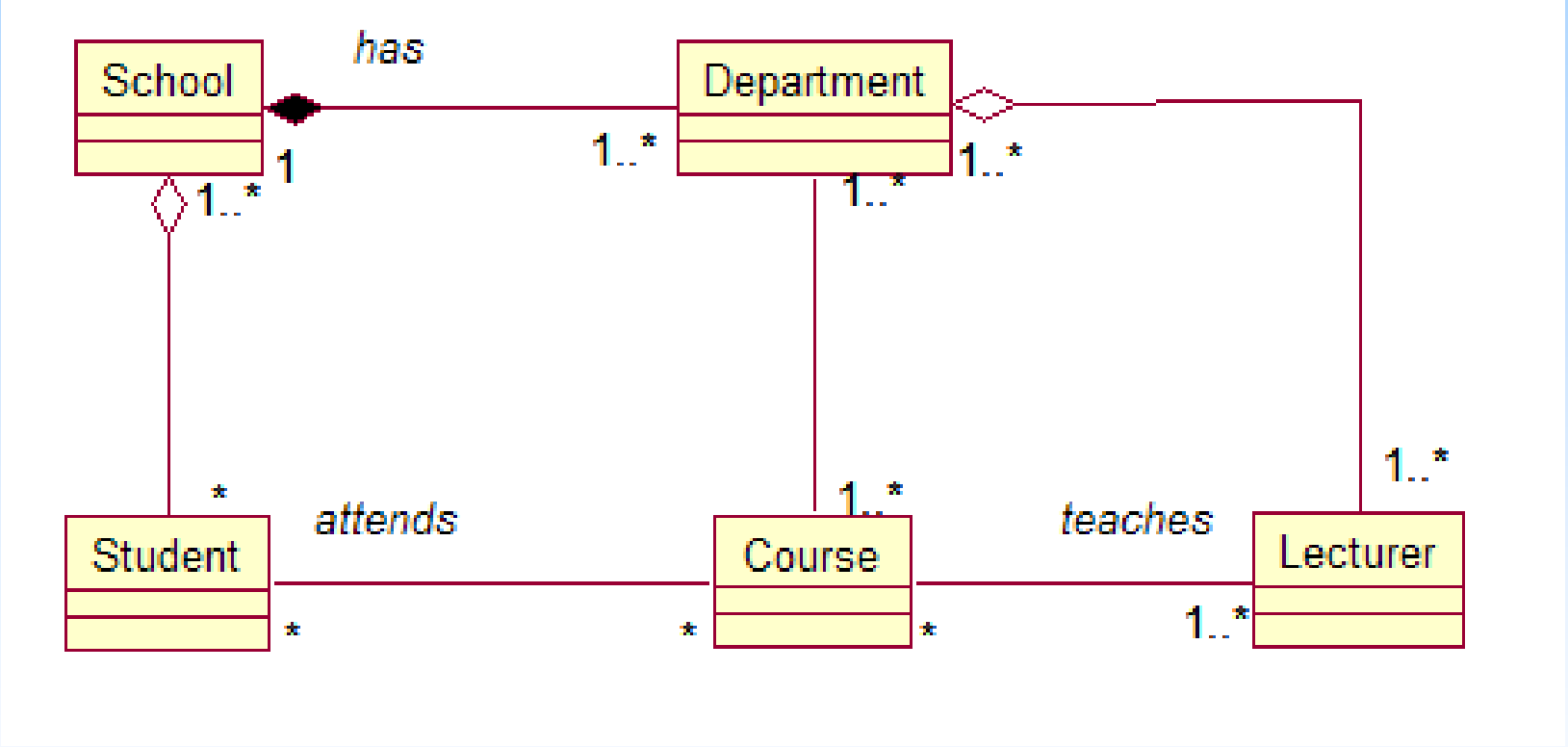
  – Shows set of *classes*, *interfaces*, and *collaborations* and their relationships.

  – Most common diagram found in modelling object-oriented system.

  – Address the static view of a system.

  class: a category or group of things that have the same attributes and the same behaviours.

UML Diagrams



Class Diagram

e.g. School of Computing

UML Diagrams

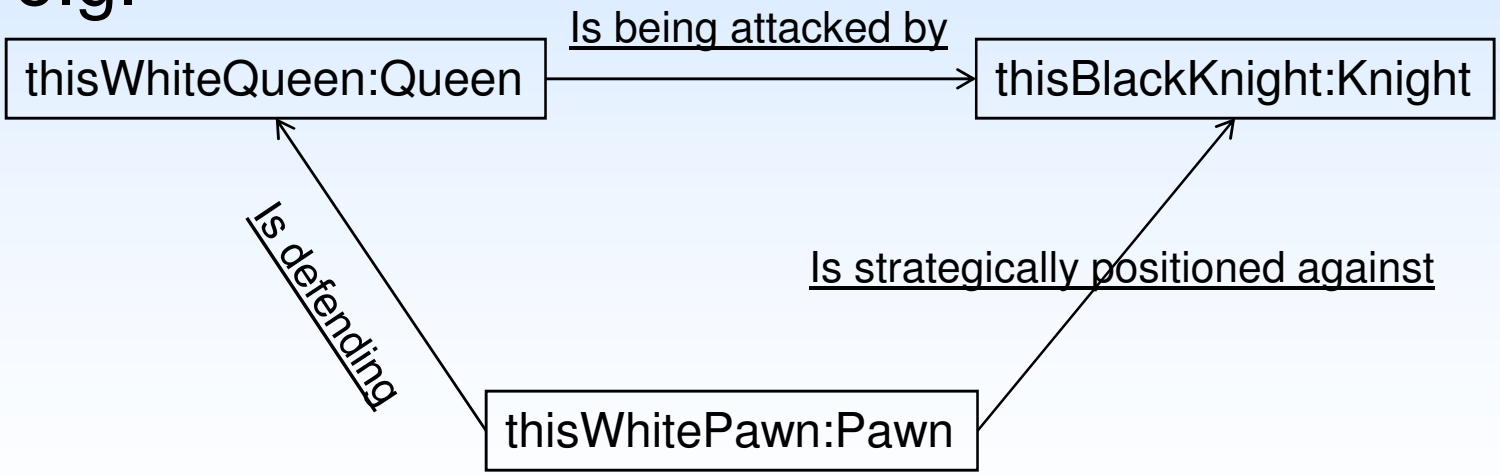
Object Diagrams :

- Similar to a class diagram
- Models actual object instances with current attribute values.
- Shows a set of *objects* and their relationships.
- Provides a snap shot of the system's object at one point in time.

UML Diagrams

Object Diagram

e.g.



UML Diagrams

Use Case Diagrams :

– Shows a set of *use cases* and *actors* and their relationships.

Use case: a description of a system's behavior from a user's standpoint.

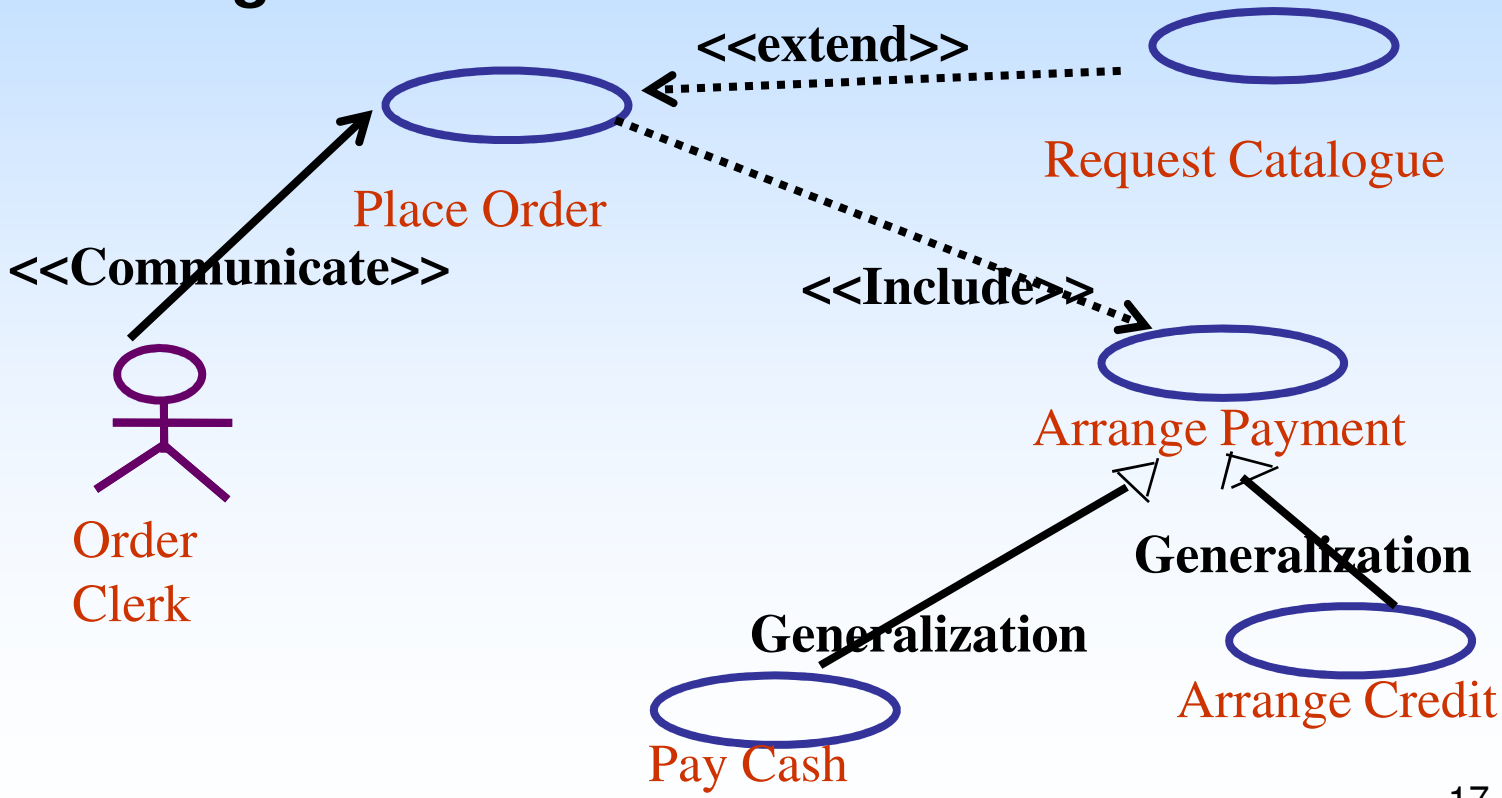
– A tried-and-true technique for gathering information.

– Graphically describes who will use the system and in what ways the user expects to interact with the system.

UML Diagrams

Use Case Diagram

Order Processing



UML Diagrams

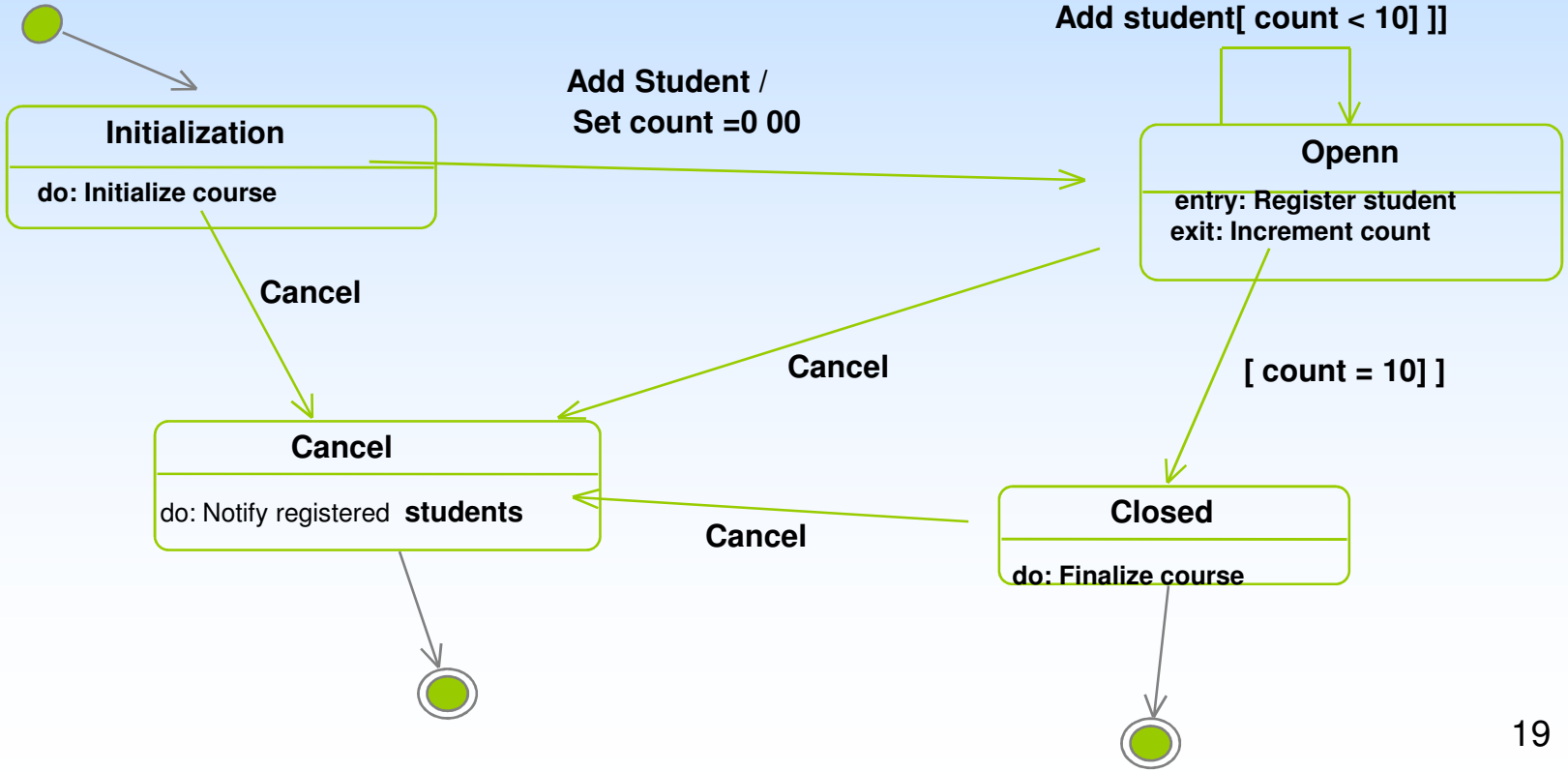
State Diagrams :

- Address the dynamic view of a system.
- Shows a state machine consisting of states, transitions, events and activities.
- Models how events can change the state of an object over its lifetime, showing both the various states that an object can assume and the transactions between those states.

UML Diagrams

State Diagrams

e.g. Course Registration



UML Diagrams

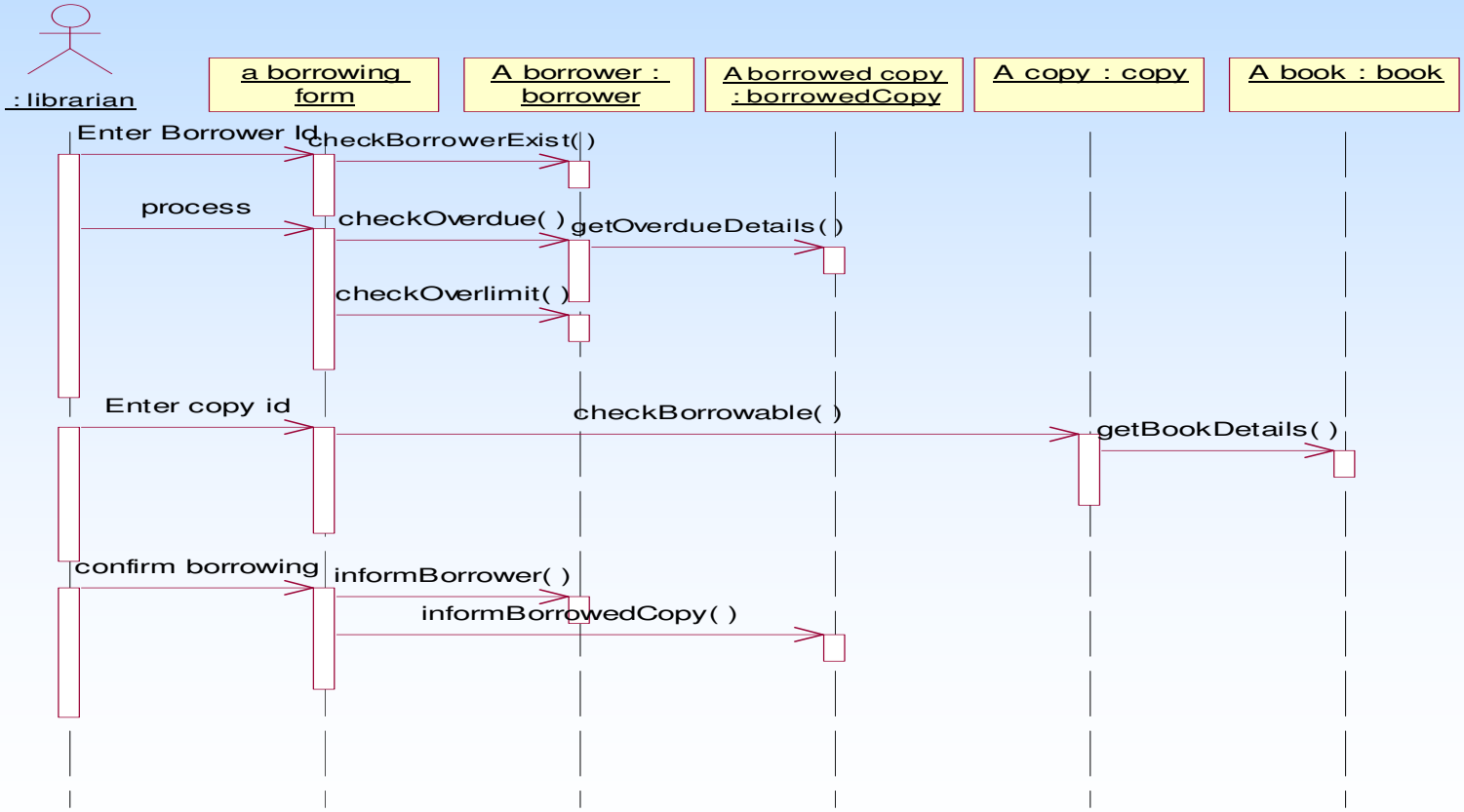
Sequence Diagrams:

- An Interaction diagram that emphasizes the time-ordering of messages.
- Graphically depicts how objects interact with each other via messages in the execution of a use case or operation.
- Illustrates how messages are sent and received between objects and in what sequence.

UML Diagrams

Sequence Diagrams

e.g. book borrowing



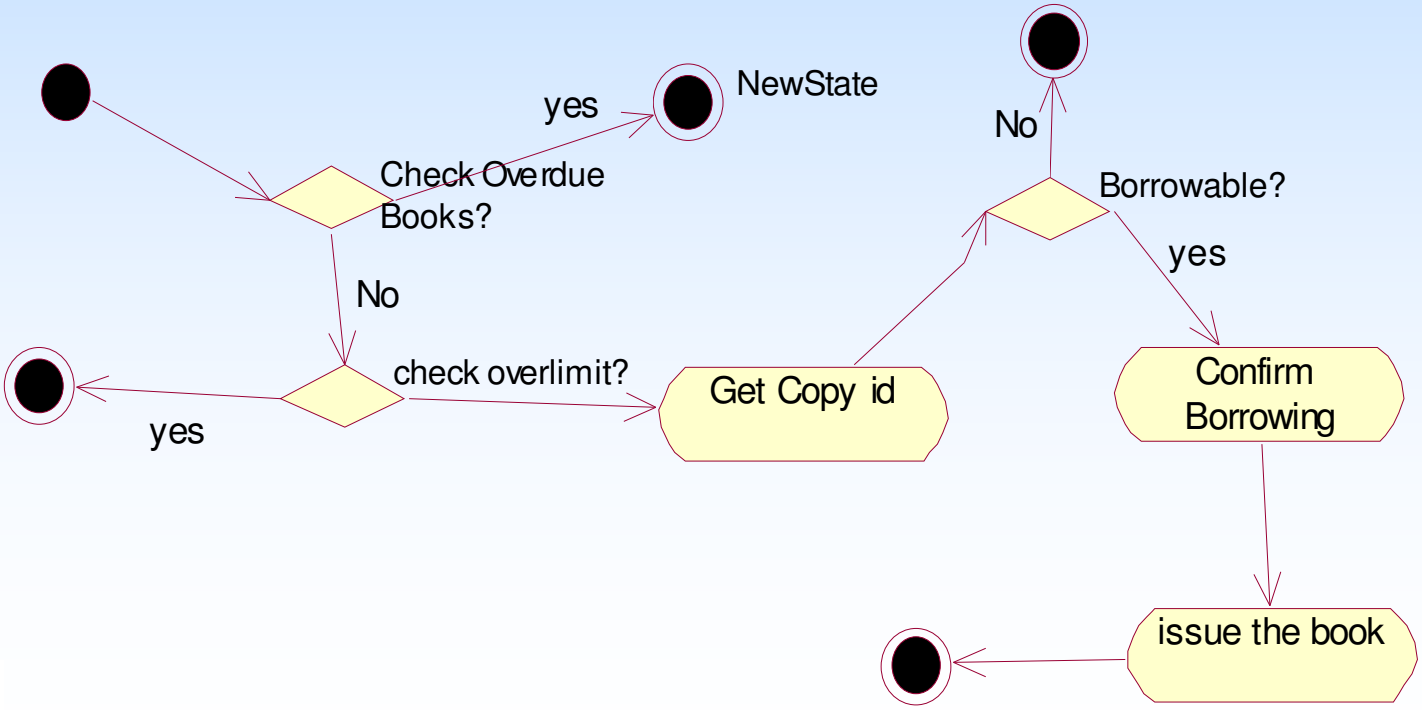
UML Diagrams

Activity Diagram :

- A special kind of a state chart diagram that shows the flow from activity to activity within a system.
- Address the dynamic view of a system.

UML Diagrams

Eg. Activity Diagram.



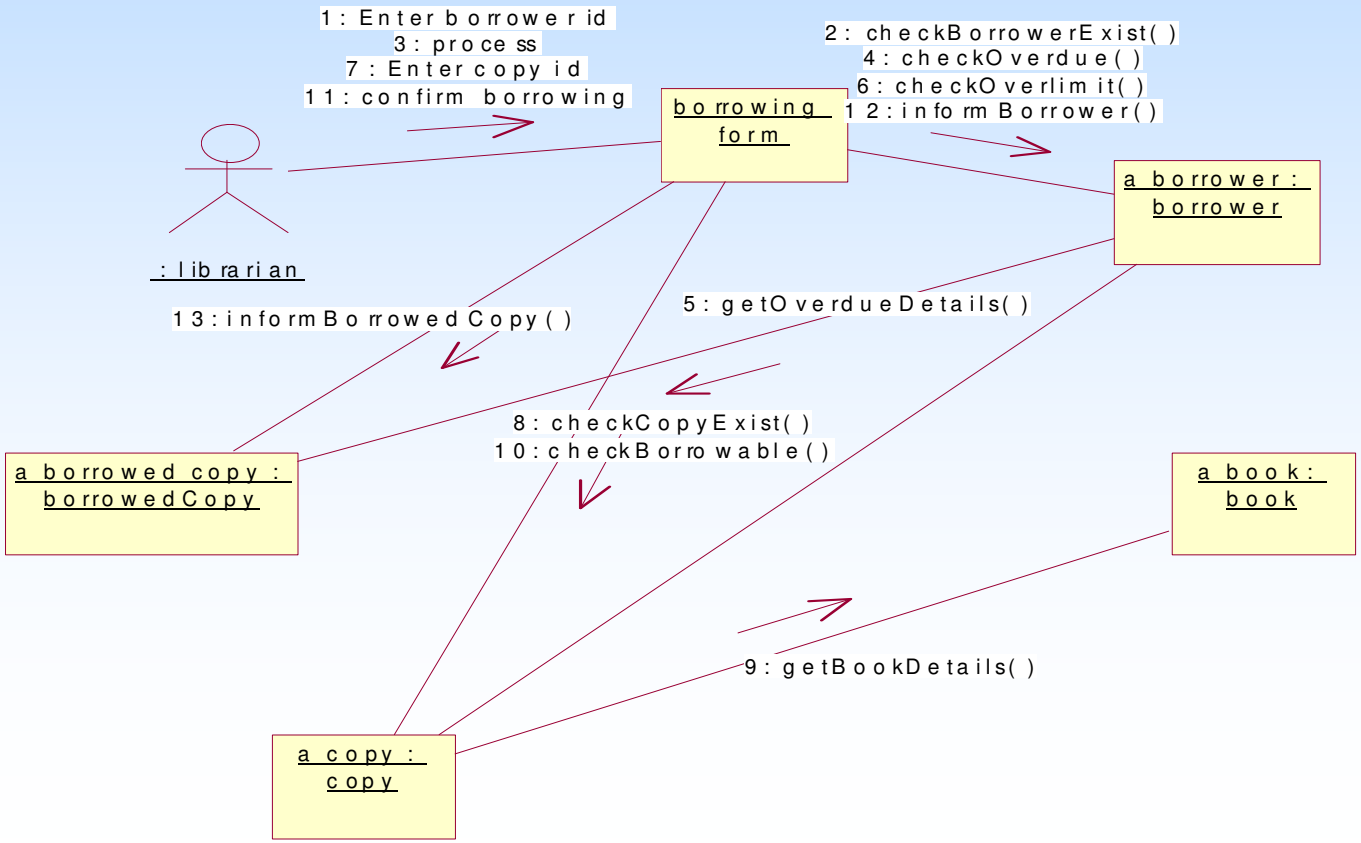
UML Diagrams

Communication Diagrams :

- An interaction diagram that emphasizes the structural organization of the objects that send and receive messages.
- Depicts the interaction of objects via messages.
- Also known as collaboration diagram in UML 1.X.

UML Diagrams

Communication Diagrams : e.g.



UML Diagrams

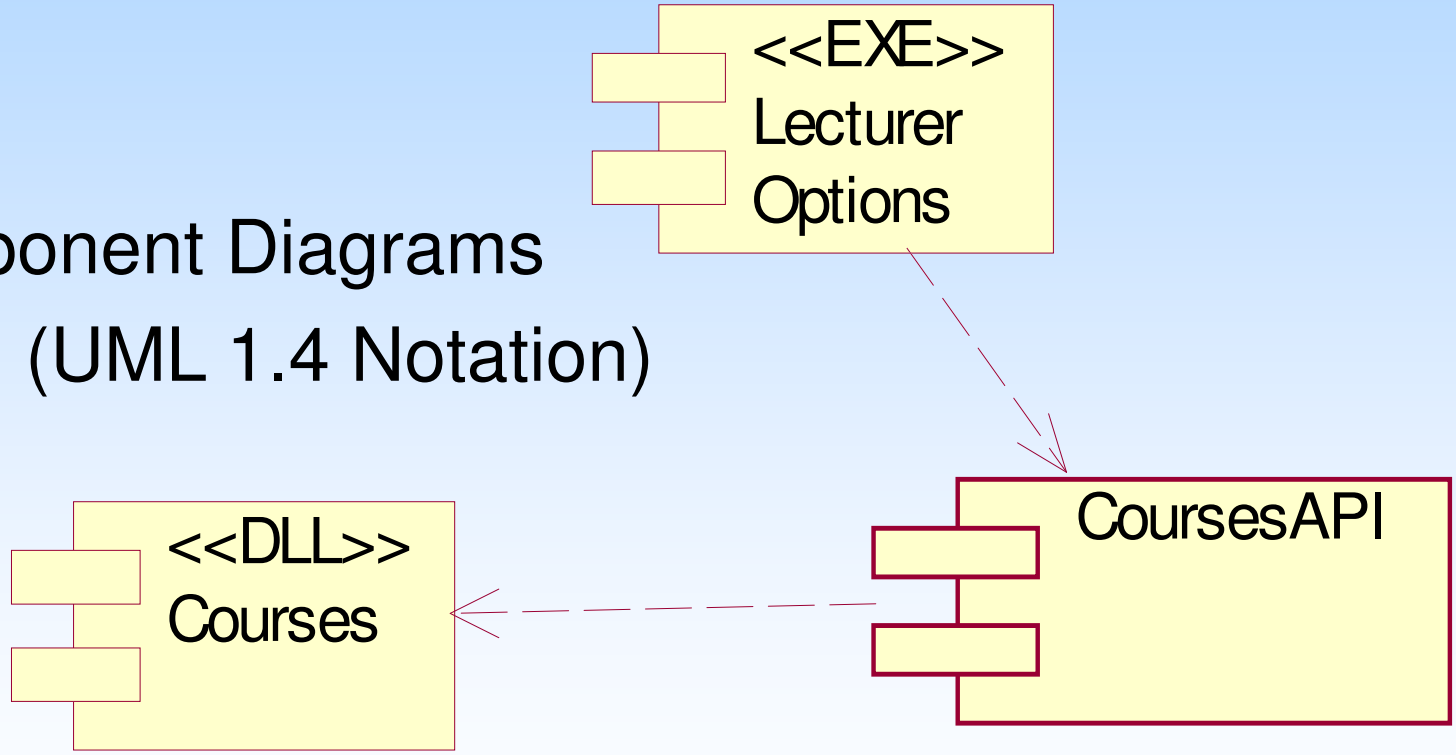
Component Diagram :

- Shows the organizations and dependencies among a set of components.
- Address the static implementation view of a system.

UML Diagrams

Component Diagrams

e.g. (UML 1.4 Notation)



UML Diagrams

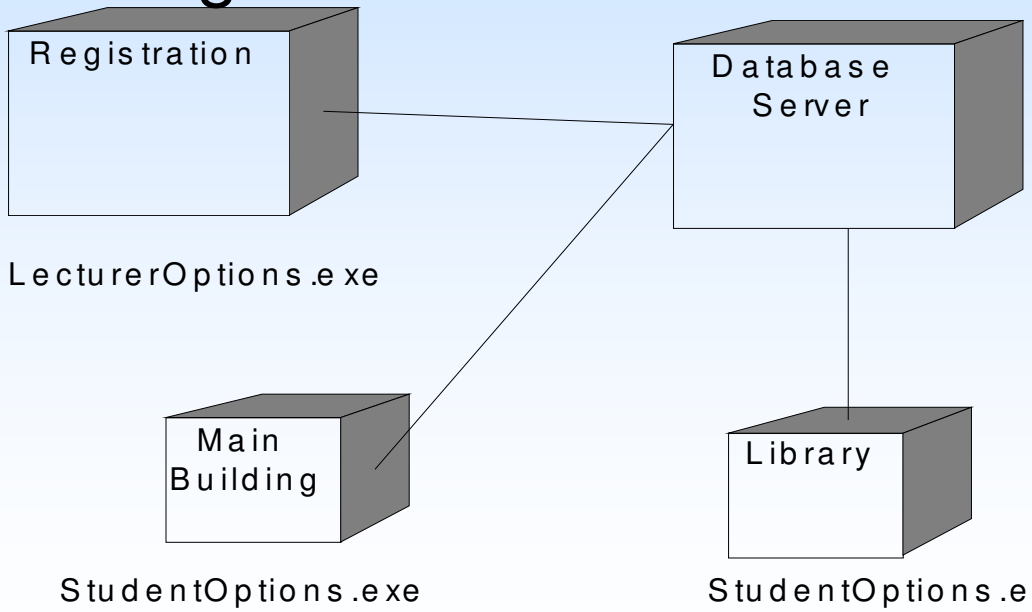
Deployment Diagram

- Shows the configuration of run time processing nodes and the components live on them.
- Shows the physical architecture of a computer based system.
- It can depict computers, their connections with one another, and show the software that sits on each machine.

UML Diagrams

Deployment Diagram

e.g.

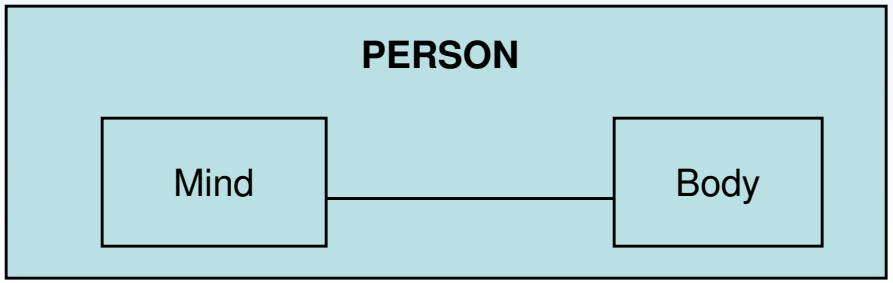


UML Diagrams

Composite Structure Diagrams :

- Decomposes the internal structure of a class.
- New in UML 2.0.
- A diagram which shows something about the class's internal structure.

e.g.

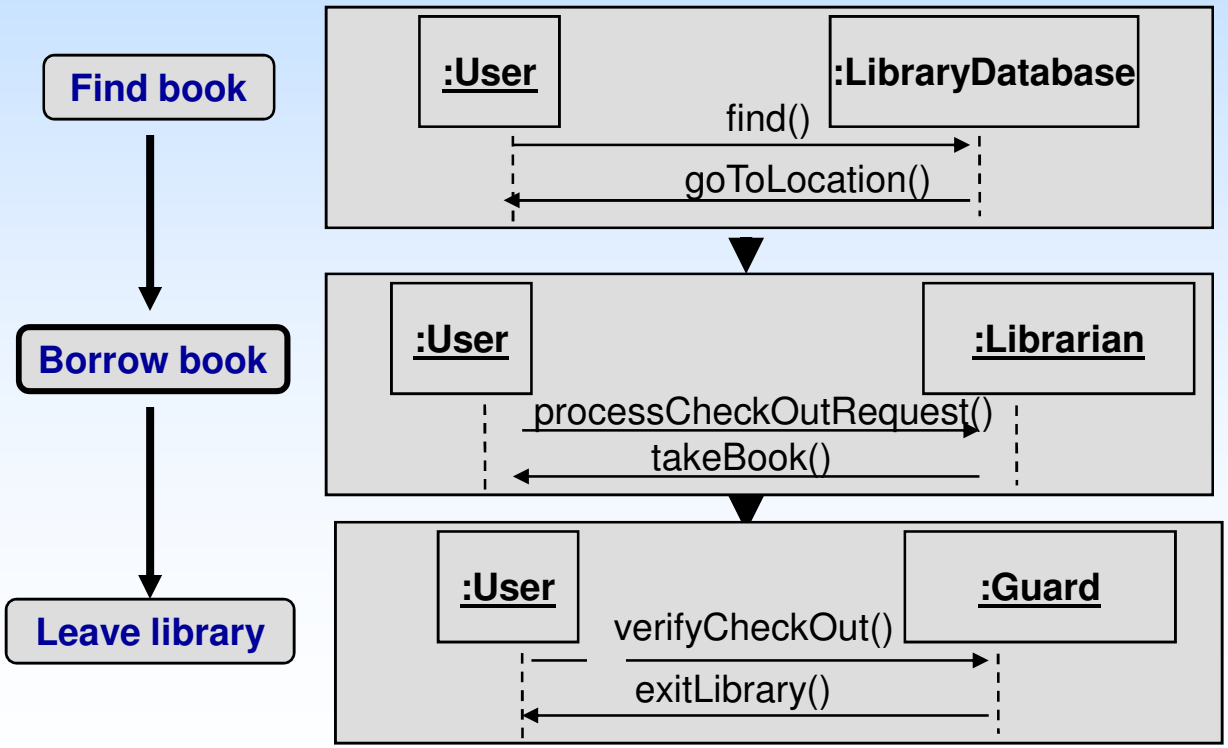


UML Diagrams

Interaction Overview Diagrams :

- New in UML 2.0.
- A variant on UML activity diagrams which overview control flow.
- Replace some of the activities in the activity diagram with sequence / communication diagrams (or a combination of the two)
- Shows how objects interact within each activity of a use case.

UML Diagrams



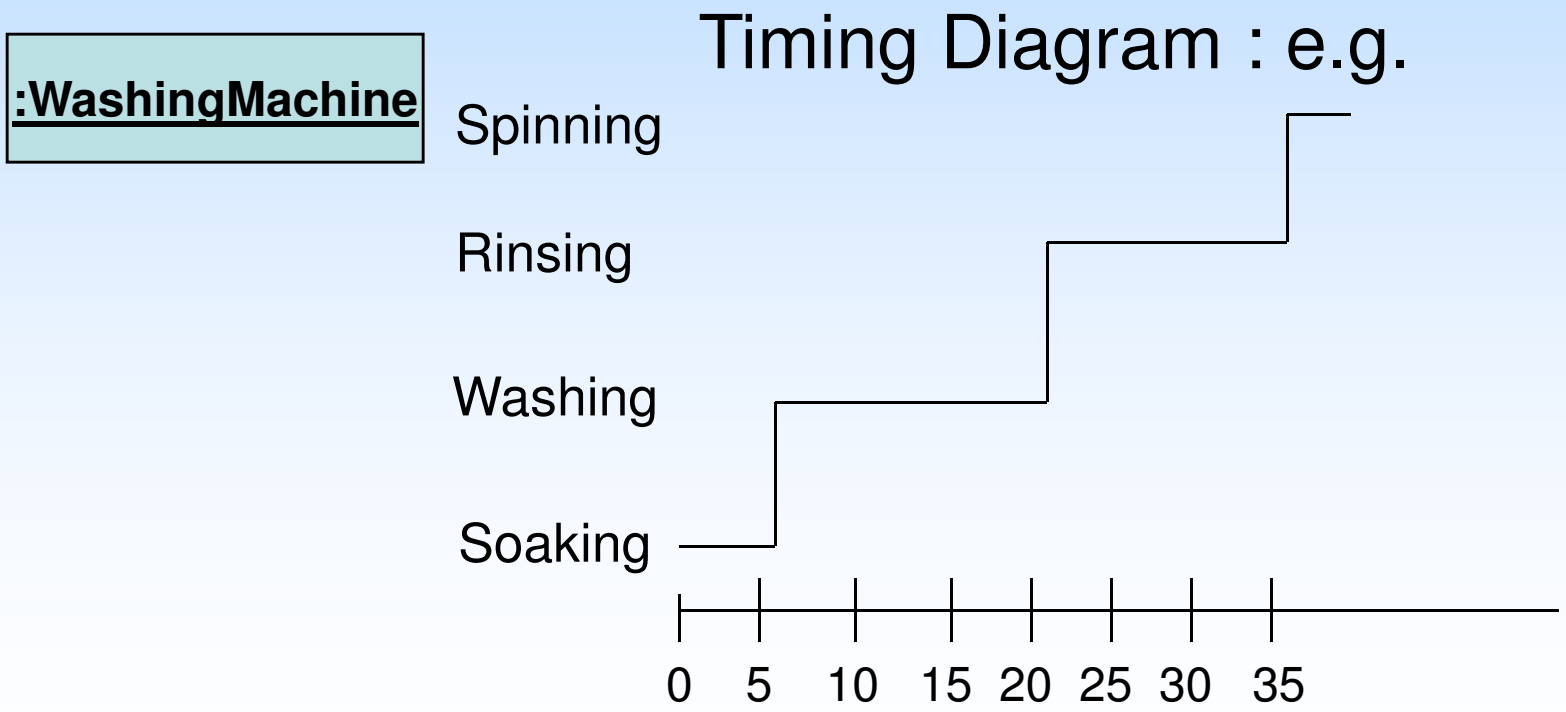
Interaction Overview Diagram e.g.

UML Diagrams

Timing Diagrams :

- New in UML 2.0.
- A diagram which shows how long an object is in a particular state.
- Specially useful when designing embedded software for devices.

UML Diagrams



UML Diagrams

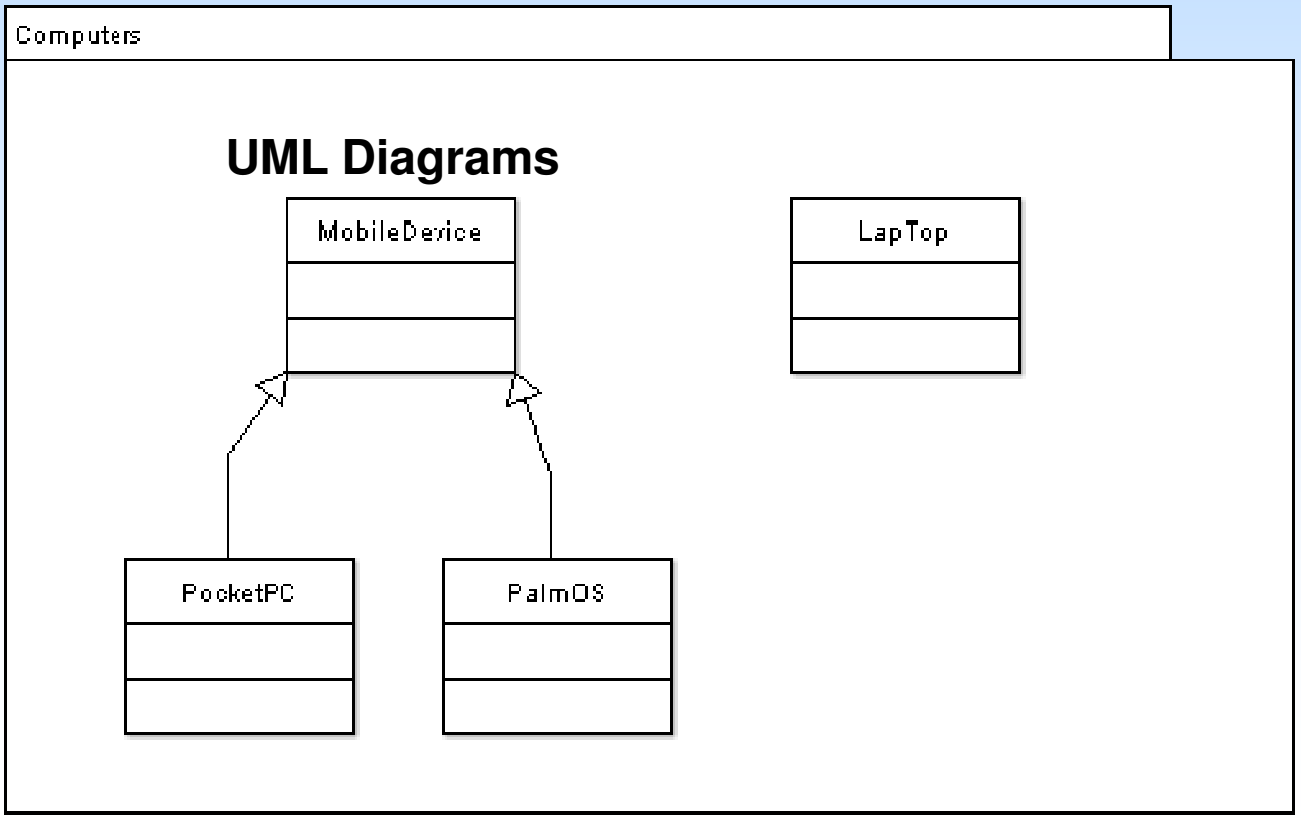
Package Diagrams:

- A diagram which combines a number of classes or components into a subsystem

UML Diagrams

Package Diagram

e.g.



UML Diagrams

Profile Diagram

Describes **lightweight extension mechanism** to the UML by defining custom **stereotypes**, **tagged values**, and constraints. Profiles allow adaptation of the UML metamodel for different:

- **platforms** (such as J2EE or .NET), or
- **domains** (such as real-time or business process modeling).

UML Diagrams

- Profile Diagram

eg.