Topic3: Object-Oriented Modeling

-Introduction to modeling
-Basics of UML modeling
-UML building blocks
-UML diagrams
-Exercises

3.1 Modeling

- Process of simplifying and representation of reality using physical or logical symbols e.g diagrams
- A model is a simplified representation of a real world object or system.

3.2 Why use models?

- to better understand systems we are developing
- to describe the structure and behavior of the system
- to experiment by exploring multiple solutions
- to document the design decisions
- to visualize the system “as-is” and “to-be”
- to provide a template or blueprint for constructing the system
- to provide abstractions for managing complexity

3.3 Principles of modeling

- No single model is sufficient to describe non-trivial system
- The choice of models will affect how a problem is tackled
- Every model may be expressed at different levels of abstraction
- Effective models are connected to reality

3.4 Basics of UML

- UML –stands for *Unified Modeling Language*
- It is a tool with conventions and notations used to describe artifacts of OOA and OOD.
- It is a language for visualizing, specifying, constructing, and documenting artifacts of software intensive systems.
- Examples of artifacts: Requirements, Architecture, Design, Source code, Test cases, Prototypes
- UML is suitable for modeling: enterprise information systems, distributed systems, realtime embedded systems

3.5 UML building blocks

– There are 3 basic building blocks of UML:

– *Elements* –basic units of UML models

–*Relationships* –connections that tie elements together in the model

–*Diagrams* –graphical representation of a collection of elements (as graph nodes) and their relationships (as graph edges)

3.6 Elements

Four basic types of elements:

1)structural –concepts that represent static parts of model i.e. ‘nouns’ of the model

2)behavioural –concepts that represent behaviorial parts of model i.e. ‘verbs’ of the model

3)grouping –concepts used to elements into logical groups i.e. ‘containers’ of elements in the model

4)annotation –concepts used to add comments or brief explanations to the model

3.61 Structural Elements

Seven kinds of structural elements:

1) class
2) interface
3) collaboration
4) use case
5) active class
6) component
7) node

1) Class

–description of a set of objects that share the same attributes, and operations

–graphically modeled as a rectangle usually including a name, attributes and operations

–types of classes:

a. concrete classes

–has methods for all operations

–methods may be: i) defined in the class or ii) inherited from a super-class

contd…

b.abstract classes

–a class that lacks a complete implementation i.e. provides operations without implementing some methods

–cannot be used to create/instatiate objects

–a concrete sub-class must provide methods for unimplemented operations

c.active classes

–a class whose objects can initiate actions internally i.e. unlike normal classes whose objects' actions are triggered externally

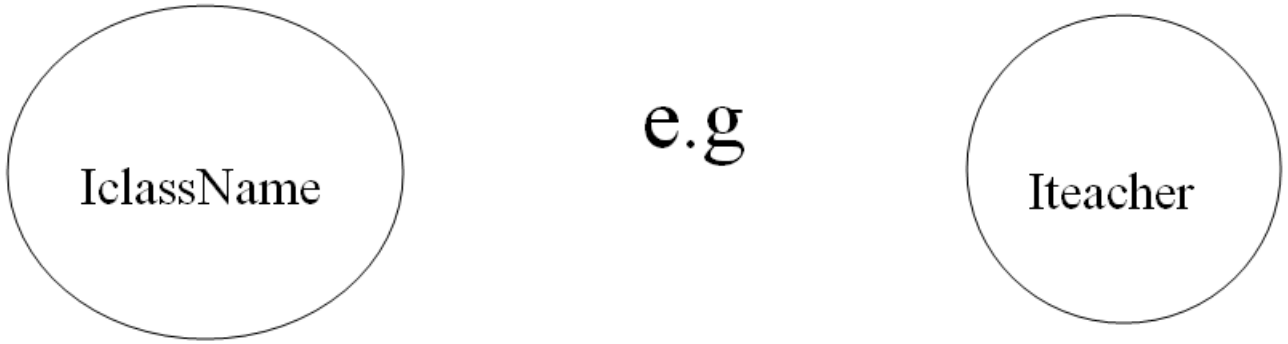
–methods of these classes implement threads

2) Interface

- Collection of operations that specifies externally visible behaviour/service of a class
- Defines a set of operation signatures but not their implementations (methods)
- modeled as:

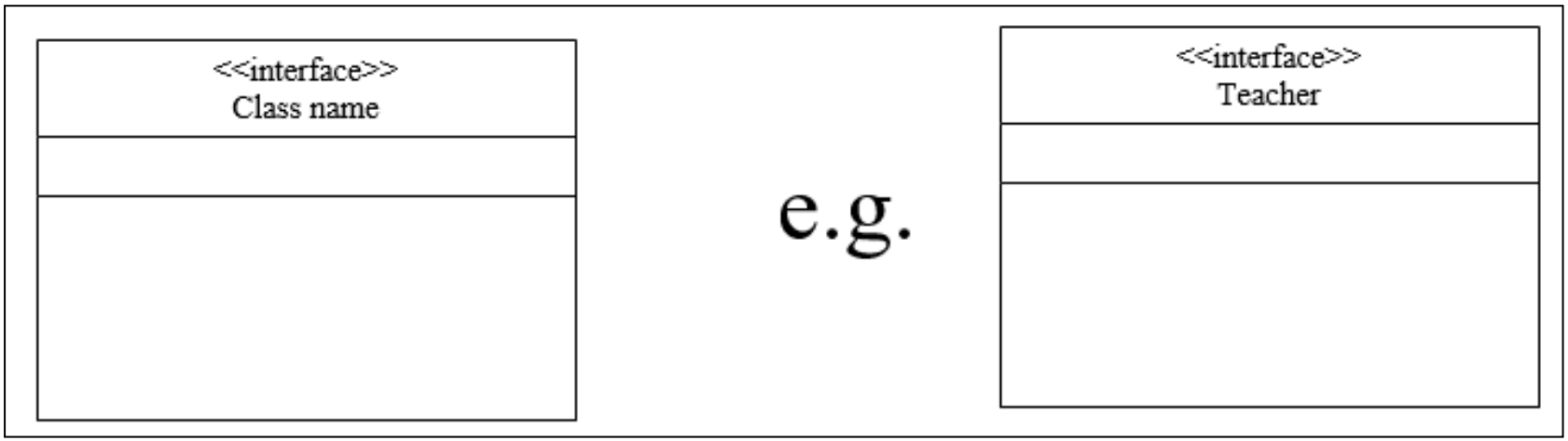
  Style1:

  a circle with a name that reflects the name of the class to which the interface belongs and prefix I i.e. IclassName



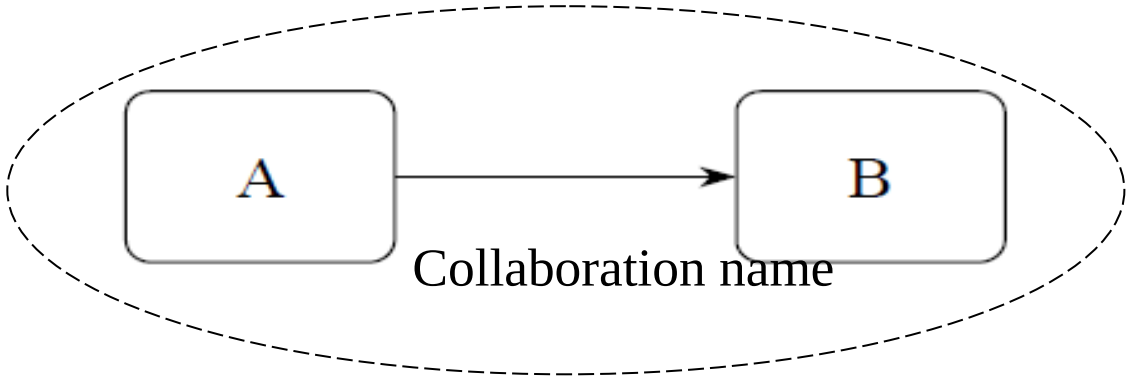
Style2:

a class (rectangle) with a name that reflects the name of the class to which the interface belongs and a stereotype <<interface>> above class name



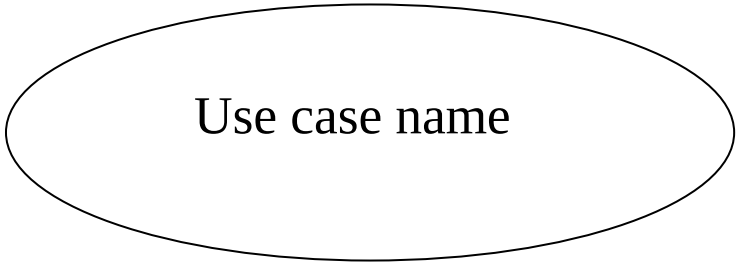
3) Collaboration

- An interaction between several elements co-operating to deliver a joint behavior.
- Graphically modeled as a named ellipse drawn with a dashed line.



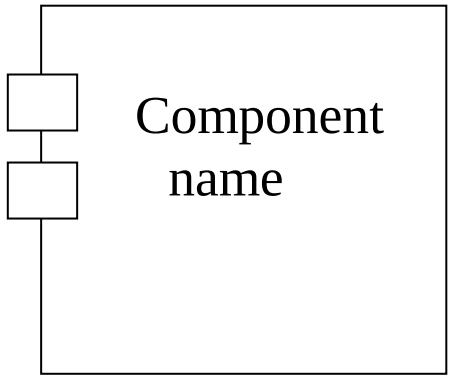
4) Use case

- Description of actions that a system performs to deliver an observable result to a particular user.
- Actions that result into observable behavior i.e. may be realized by collaboration
- Graphically modeled as a named ellipse drawn with a solid line



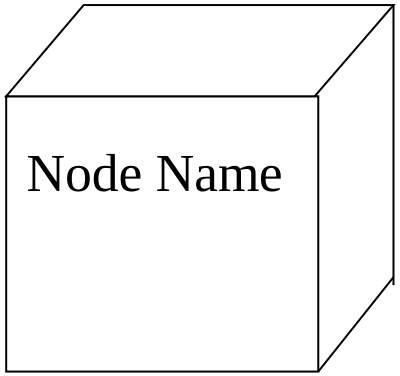
5) Component

- A physical packaging of logical elements (e.g. classes, interfaces) implemented as physical and replaceable parts of a system that are deployable such as Java beans, database, server app, client app, library, file, webpage etc.
- Graphically modeled as a named rectangle with a pair of protruding teeth on one side



6) Node

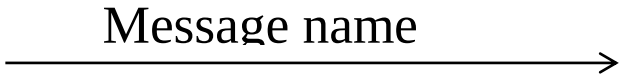
- Physical element with resources (e.g. memory, storage, processor) that provide computational support for components at run time.
- A node houses components which may also migrate from one node to another
- Graphically modeled as a named cube



3.62 Behavioral Elements

Two kinds of behavioural elements:

1) Interaction
   - –exchange of *messages* among a set of *linked* objects within a particular context to accomplish a specific goal or *behavior*
   - –graphically modeled as an arrowed line with a message

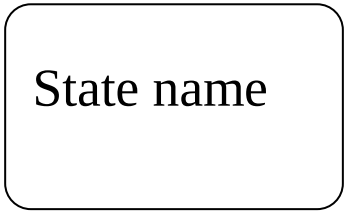


2) state machine

–a sequence of states an object undergoes during its lifetime of interaction

–includes a number of concepts: states, transition, events and activities

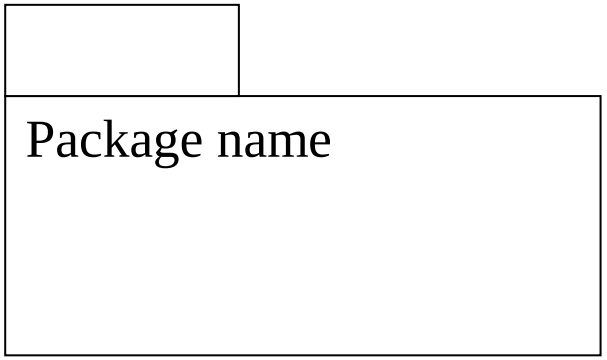
–states are graphically modeled as a rounded rectangle with names of state



3.63 Grouping Elements

Only one kind of grouping element:

1) Package
   - –organizes elements into groups
   - –graphically modeled as a named tabbed folder



E.G.

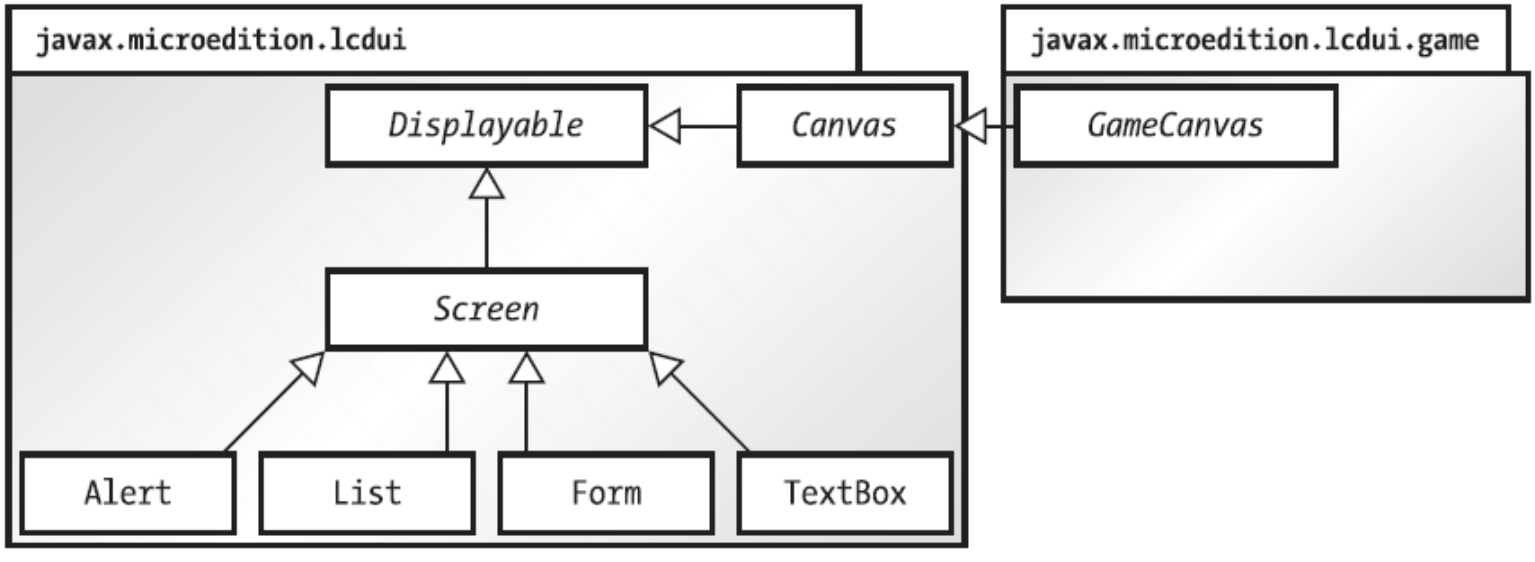
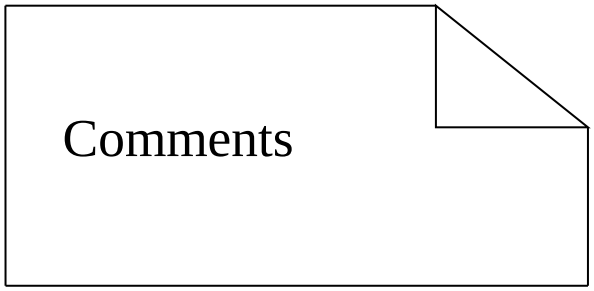


Figure 5-1. Displayables in the `javax.microedition.lcdui` *and* `javax.microedition.lcdui.game` *package*

3.64 Annotation Elements

Only one kind of grouping element:

1) Note
   - –adds comments or brief explanations to models
   - –graphically modeled as a rectangle with a dog-eared corner with comments inside



EXERCISE

Identify the UML element which best models each of the following entities:

1) Jackie Chan
2) a set of operations for validating dates
3) a module capable of validating dates based on the operations in 2
4) a set of trainees
5) “withdraw money” request from an ATM system
6) a computer server that hosts the payroll system

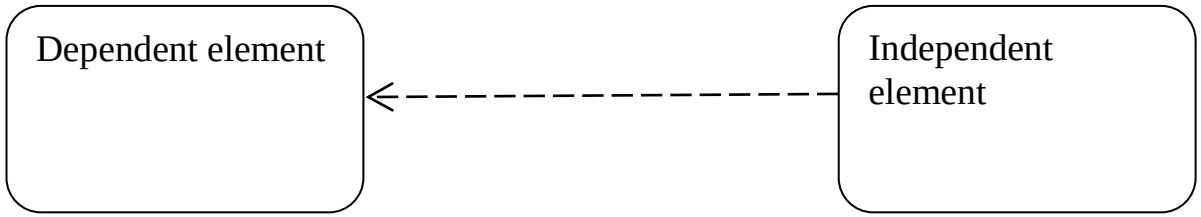
3.7 Relationships

–Connections between elements that help to bind them together for a specific purpose e.g inheritance, collaboration, or interaction.

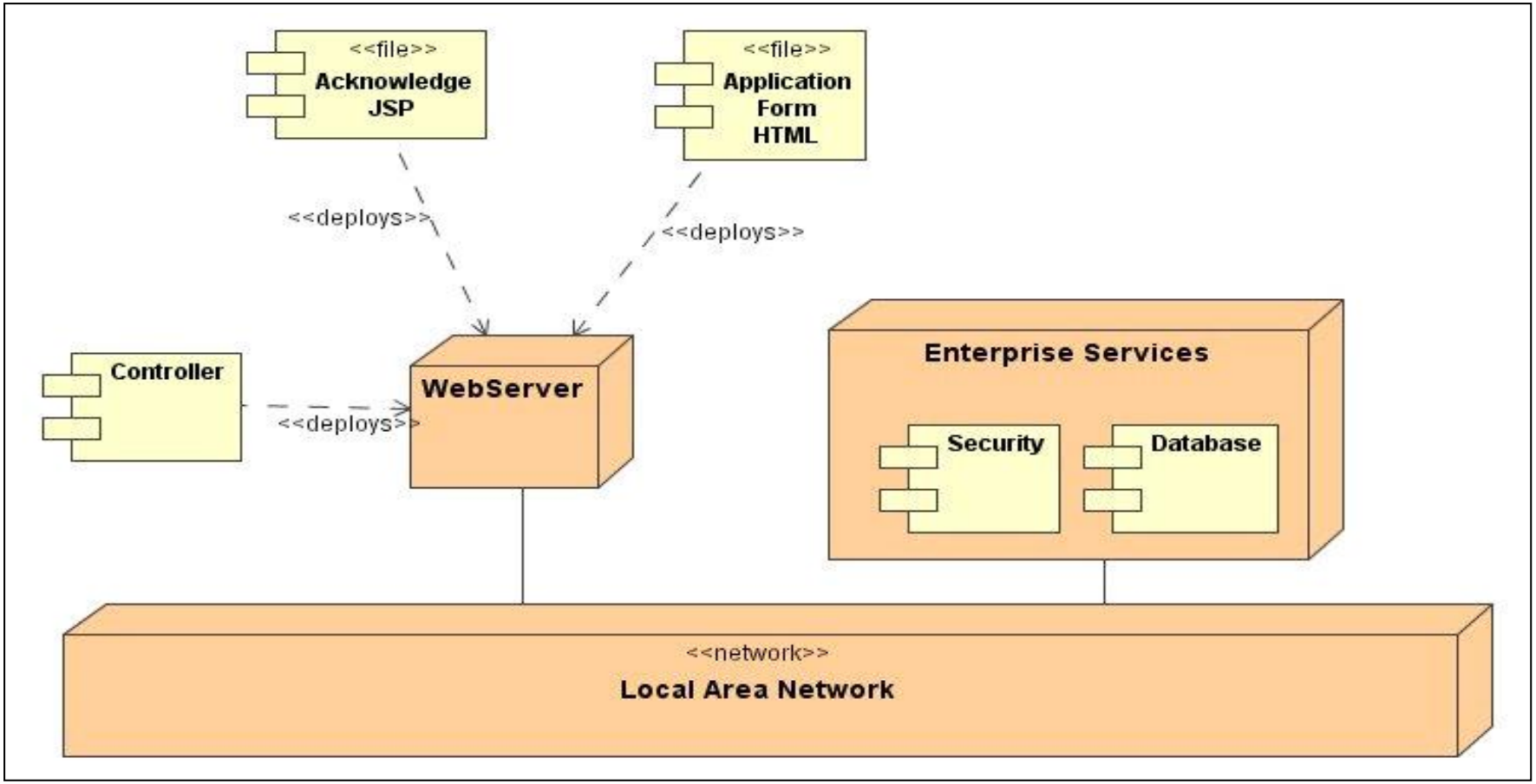
–Six basic types of relationships:

1) dependency
2) association
3) aggregations
4) composition
5) generalization
6) realization

1) dependency
- A semantic relationship between two elements in which a change to one element (independent element) may affect the meaning of another (dependent element).
- modeled as a directed dashed line possibly with a label:



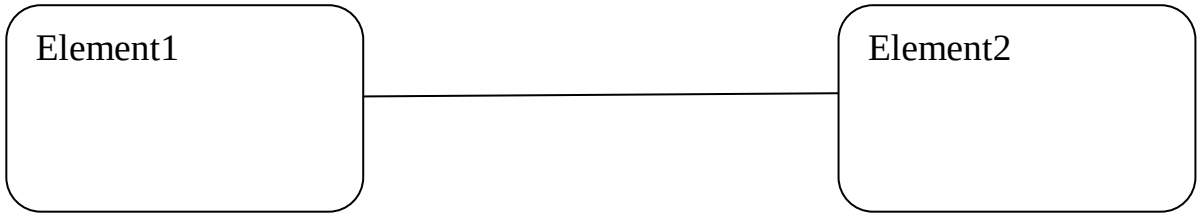
E.G.


<<file>>
Acknowledge JSP
<<file>>
Application Form HTML
<<deploys>>
<<deploys>>
Controller
<<deploys>>
WebServer
Enterprise Services
Security
Database
<<network>>
Local Area Network

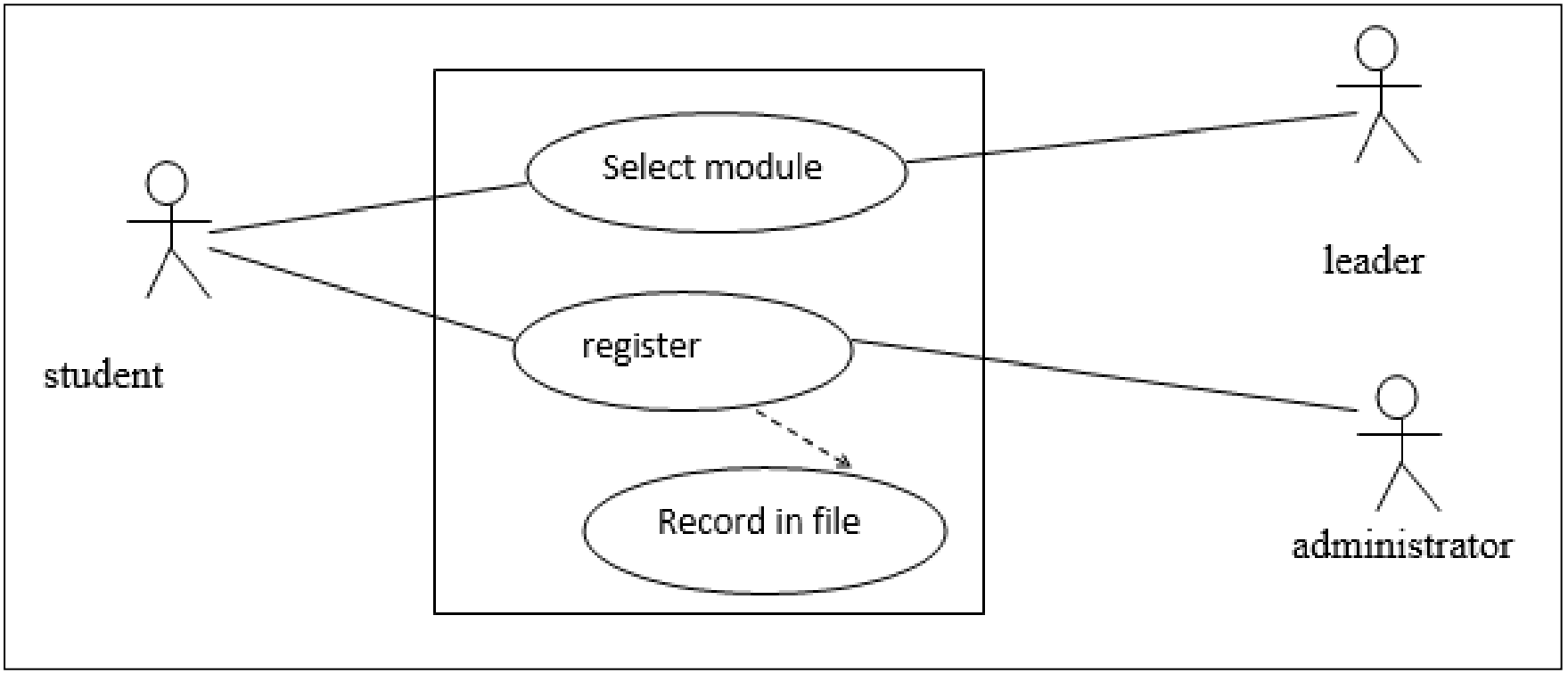
2) association

–A data-oriented relationship between two elements that signals one uses the other.

–modeled as a solid line possibly with a role label and multiplicity:

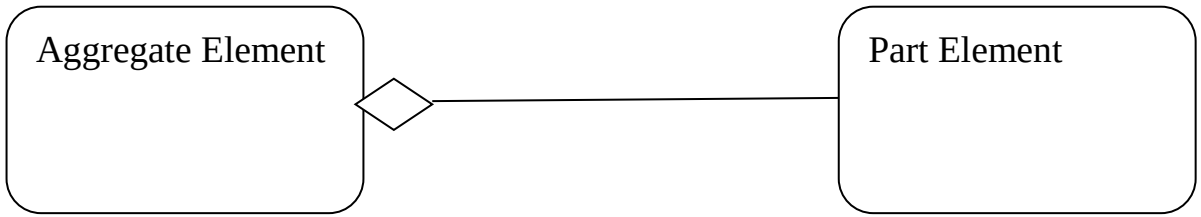


E.G.

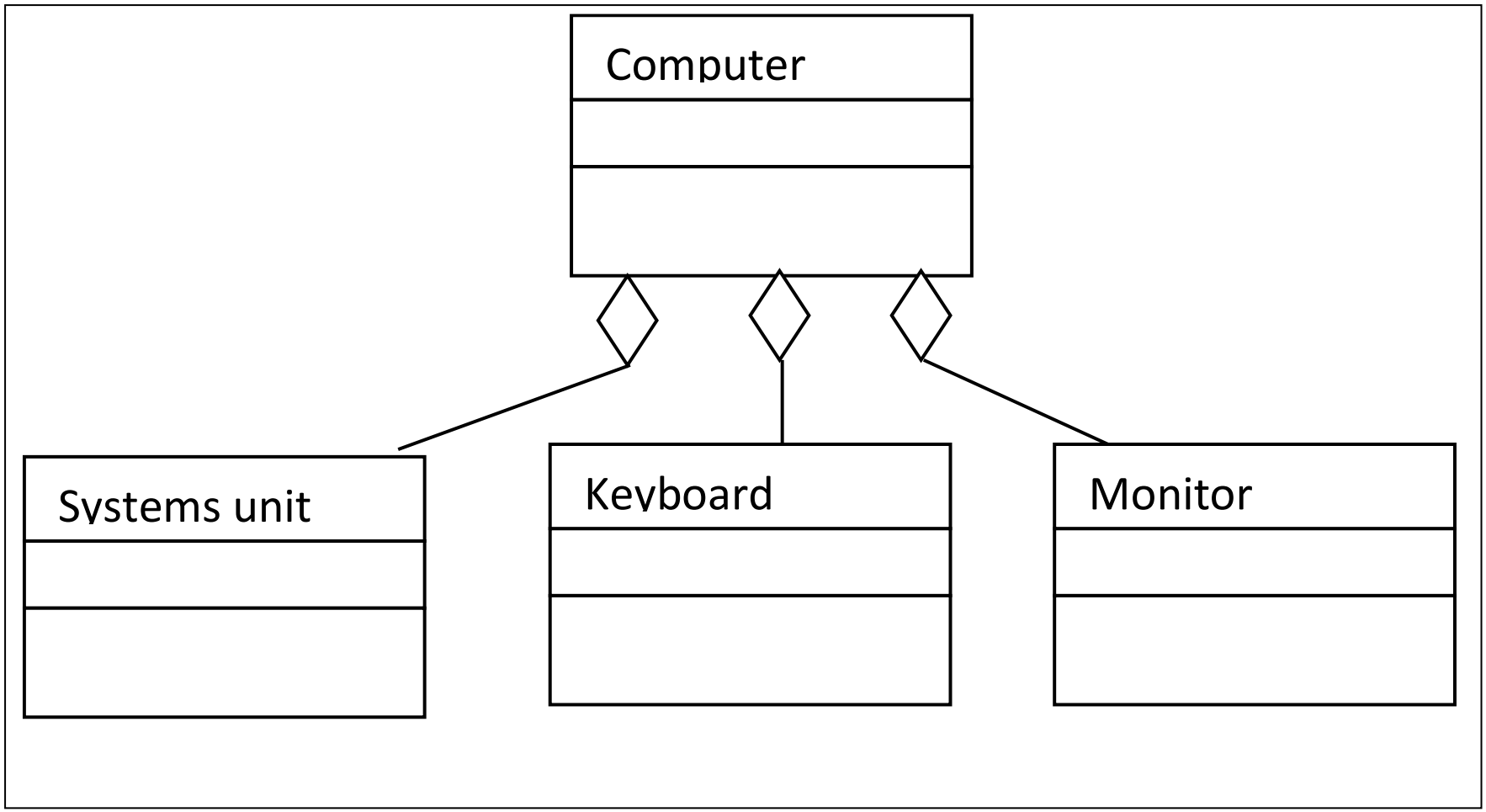

Select module
register
Record in file
student
leader
administrator

3) aggregation

- A structural relationship between independent elements that signals a whole-part relationship where part elements assemble to form the whole (aggregate element).
- modeled as a solid line with a hollowed diamond head pointing to the aggregate element:



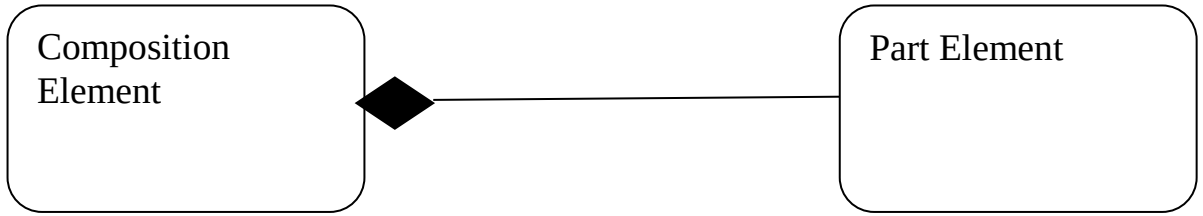
E.G.



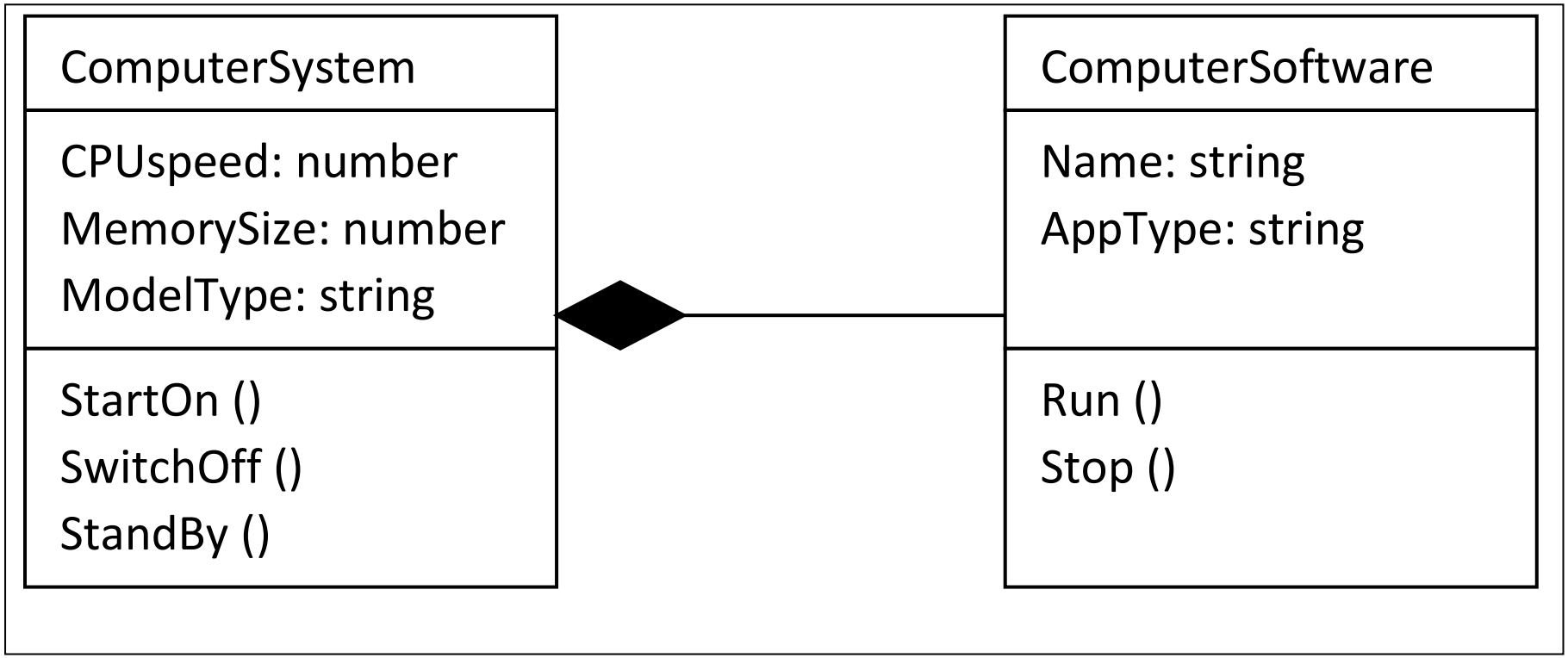
4) composition

– A structural relationship between dependent elements that signals a whole-part relationship where part elements join to form the whole (composition element).

– modeled as a solid line with a shaded diamond head pointing to the composition element:



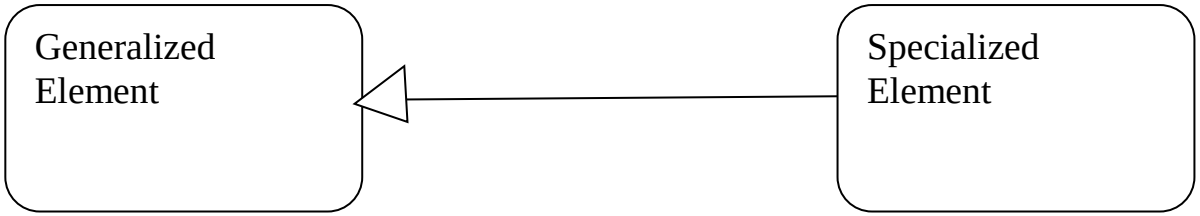
E.G.


ComputerSystem
CPUspeed: number
MemorySize: number
ModelType: string
StartOn ()
SwitchOff ()
StandBy ()
ComputerSoftware
Name: string
AppType: string
Run ()
Stop ()

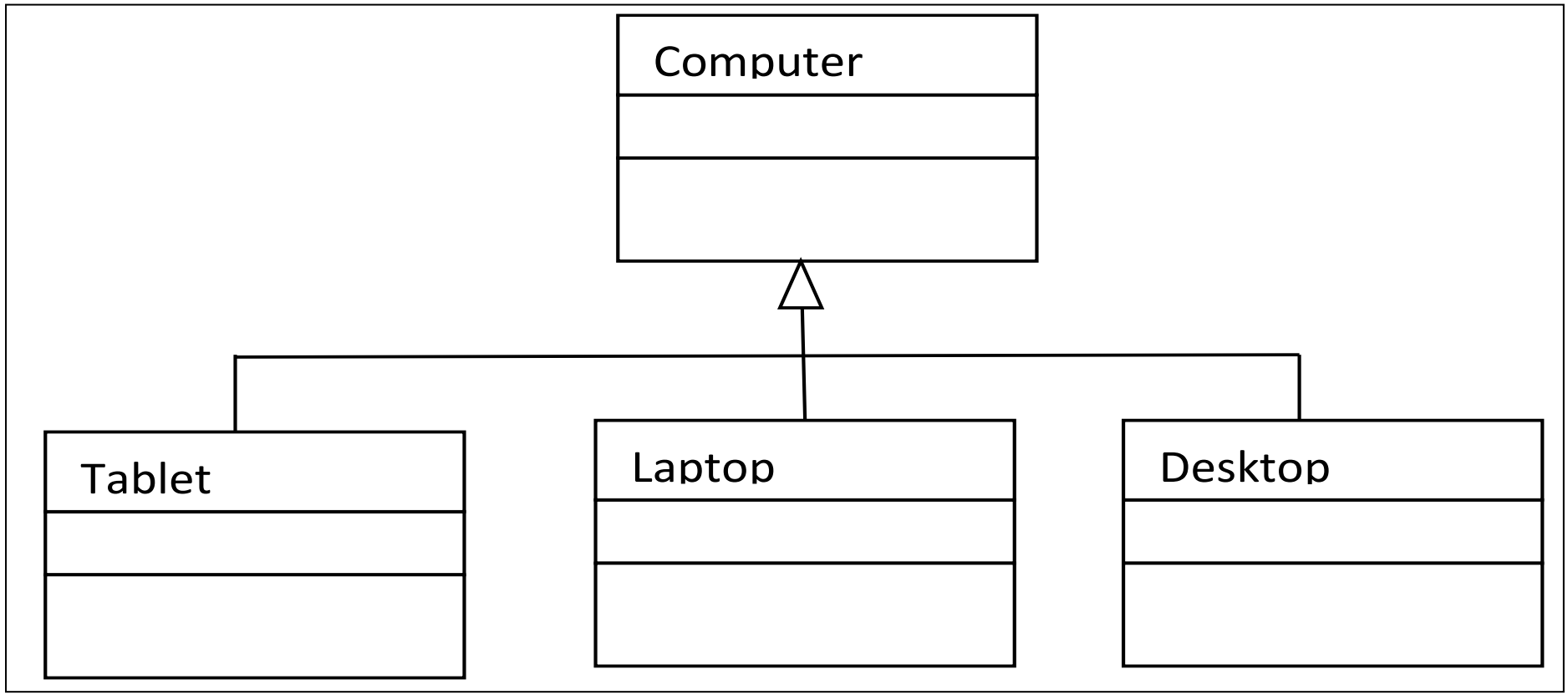
5) generalization

–a relationship in which objects of a specialized element (child) are substitutable for objects of a generalized element (parent)

–modeled as a solid line with a hollowed arrow head pointing to the generalized element:



E.G.



6) realization

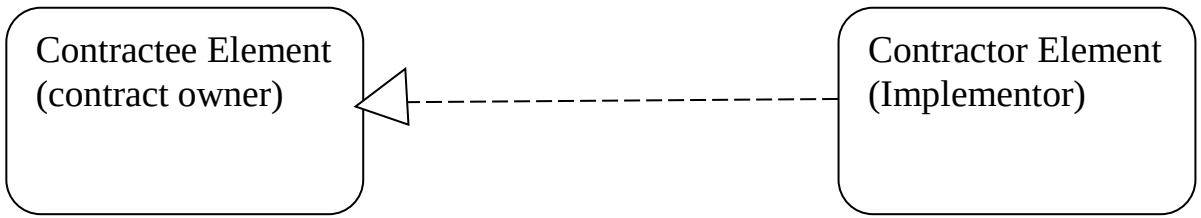
–a semantic relationship between elements, wherein one element specifies a contract and another guarantees to carry out this contract

–relevant in two basic scenarios:

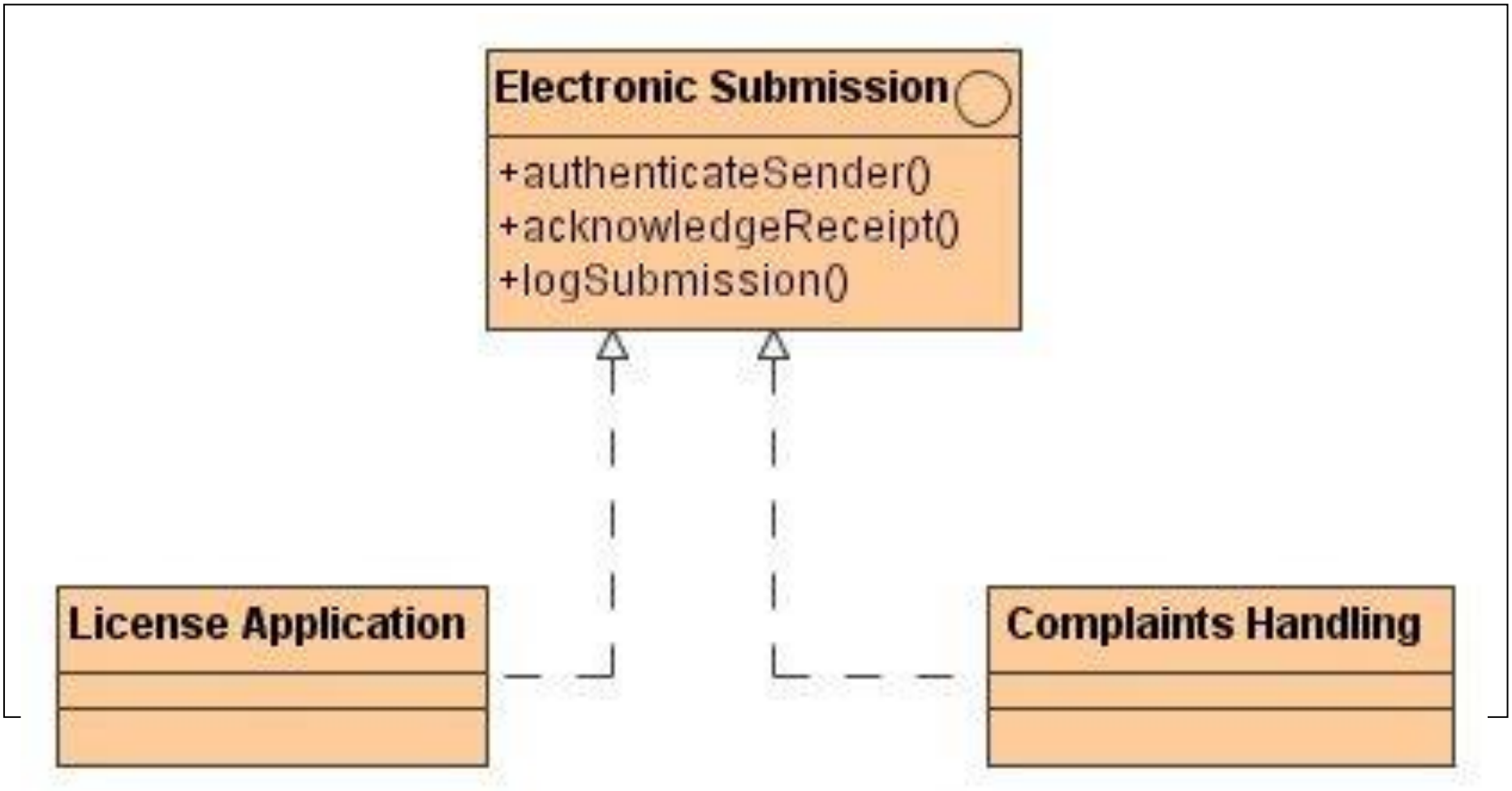
a) interfaces versus realizing classes

b) uses cases versus realizing collaborations

–graphically modeled as a dashed arrowed line with hollow head, pointing to the contractee (owner of contract)



E.G.


Electronic Submission
+authenticateSender()
+acknowledgeReceipt()
+logSubmission()
License Application
Complaints Handling

3.8 Diagrams

–A diagram is a graphical presentation of a set of elements and relationships.

–UML modeling is characterized by nine UML diagrams:1) class diagram

2) object diagram
3) use case diagram
4) sequence diagram
5) collaboration diagram
6) state chart diagram
7) activity diagram
8) component diagram
9) deployment diagram

3.81 Types of UML models

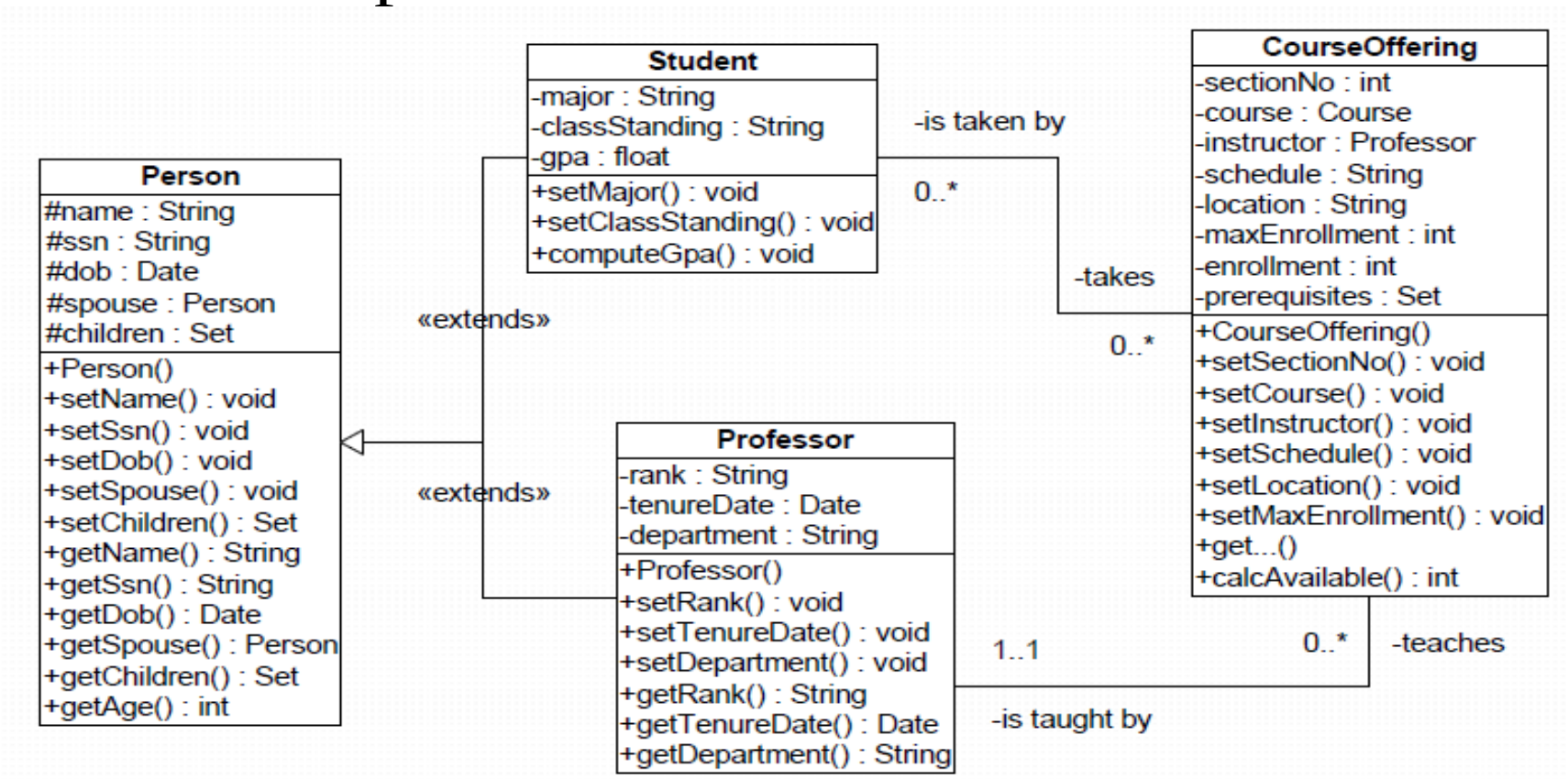
–UML diagrams are used to create two types of models:

a)static models –represent basic aspects of the system that are stable and rarely change e.g. classes, interfaces, use cases, nodes, components e.t.c

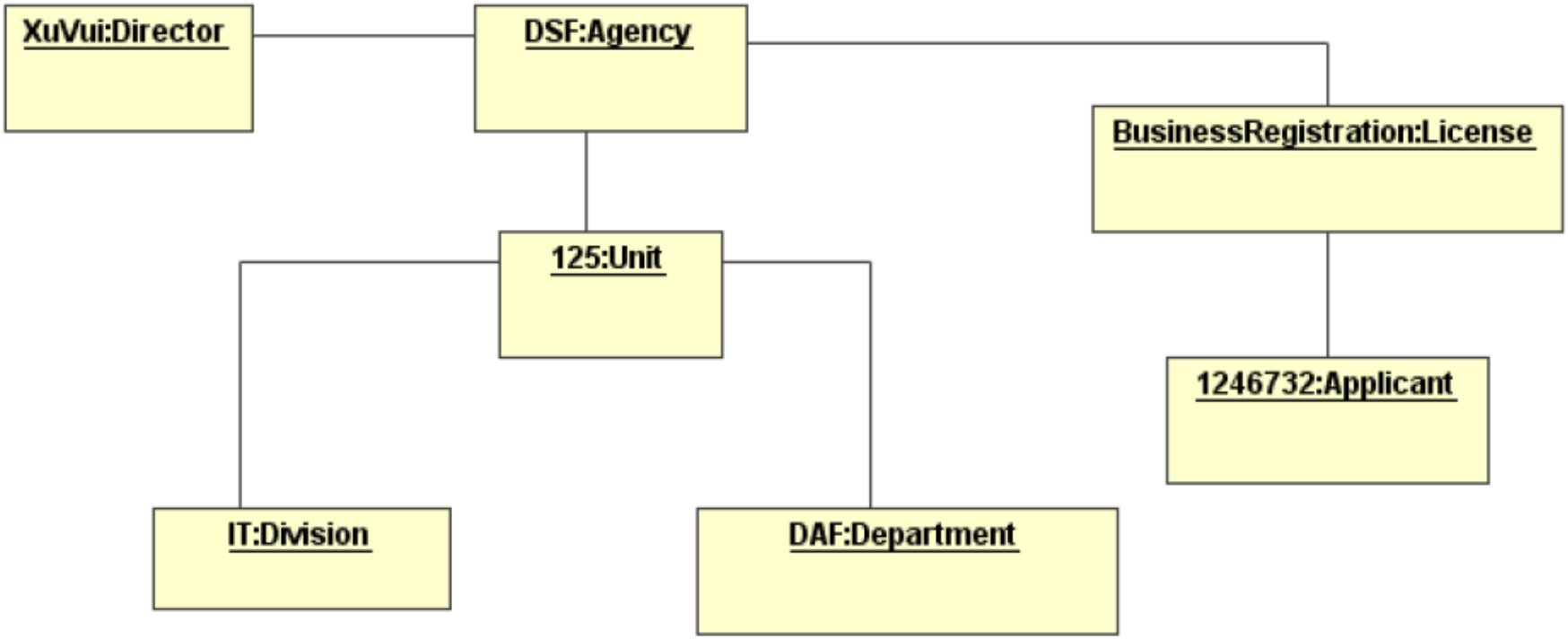
b)dynamic models –represent behaviorial aspects of the system that are constantly changing e.g. interactions between elements, states of elements, e.t.c

3.82 static modeling UML diagrams

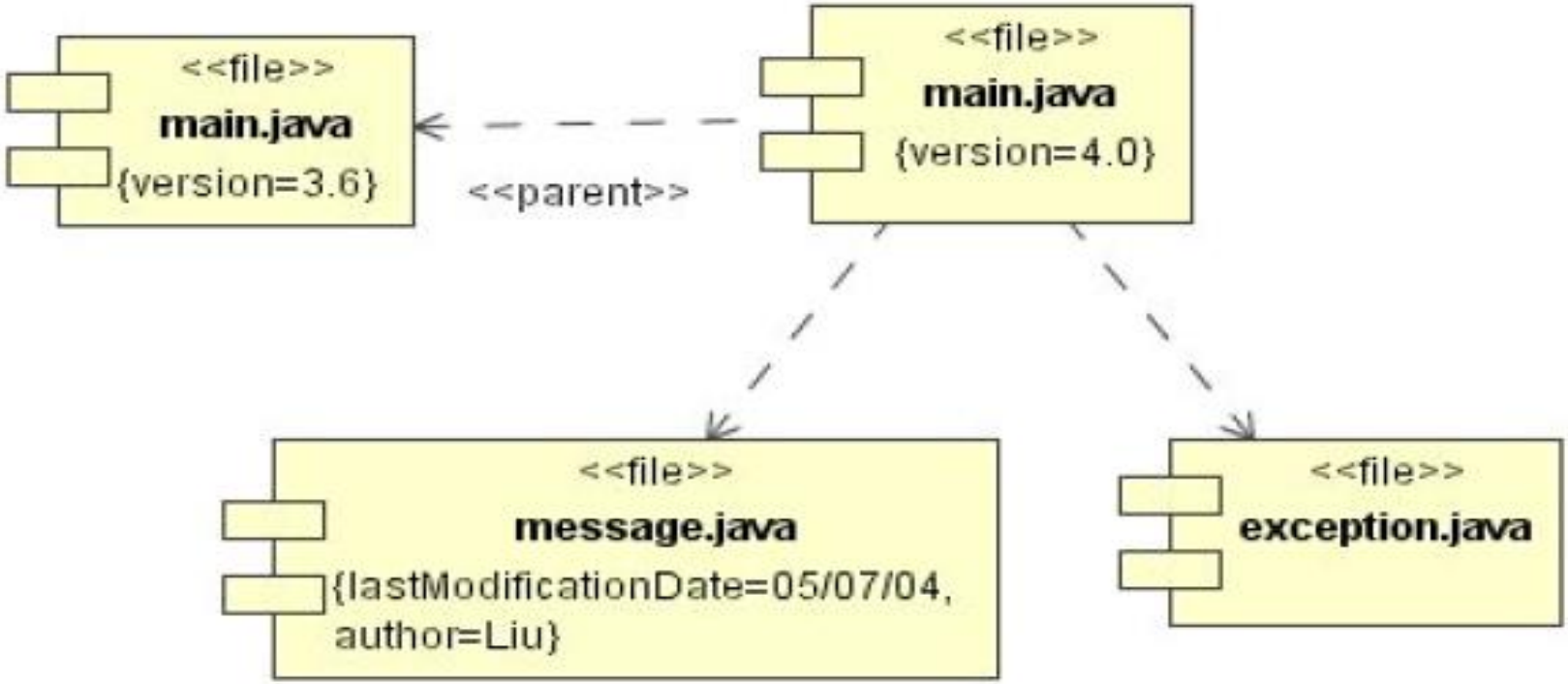
a)class diagram –graphically represents a set of classes, interfaces and collaborations, and their relationships



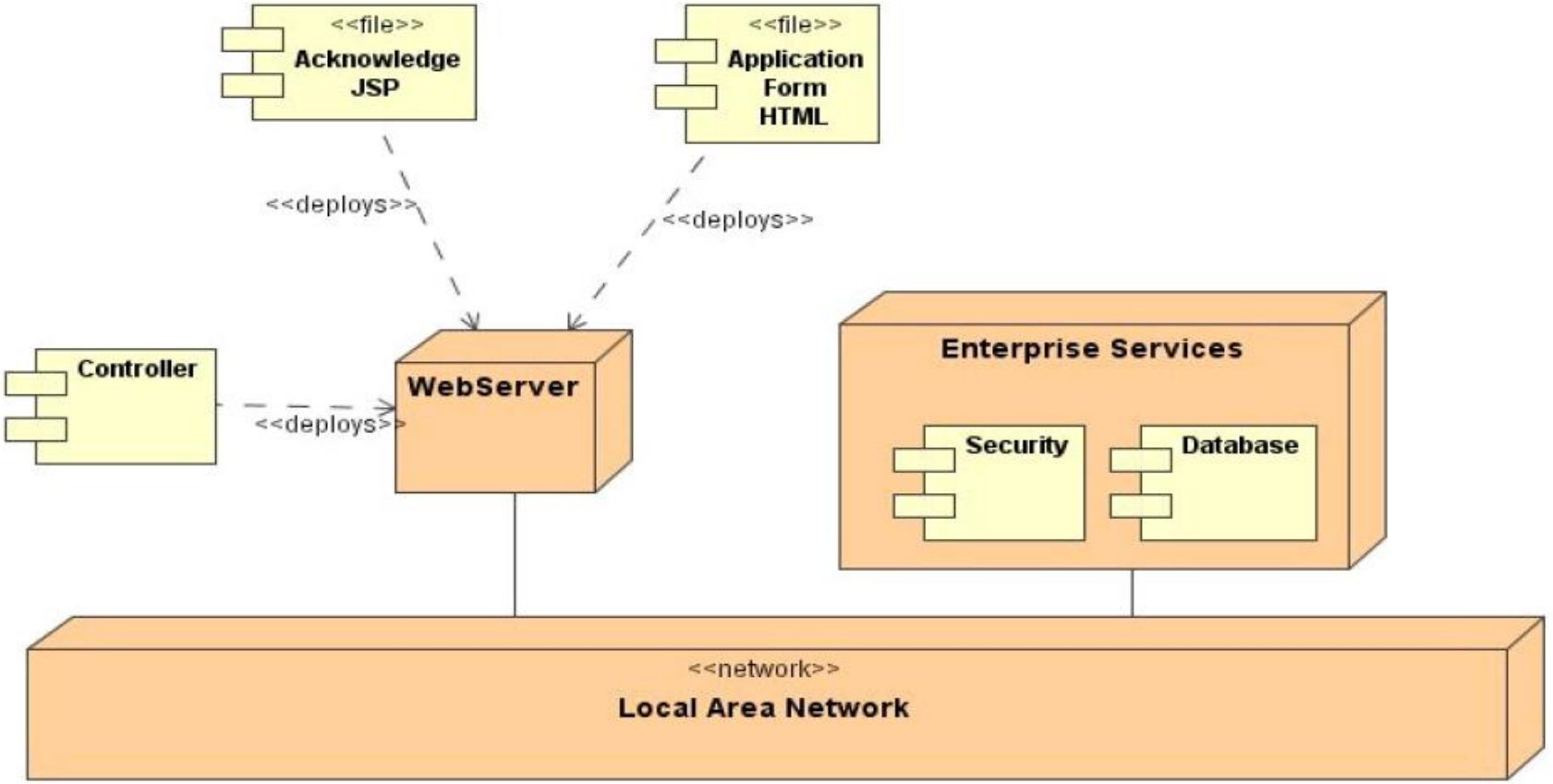
b)object diagram –graphically represents instance of classes in the class diagram and their relationships at one time.



c)component diagram –graphically represents a set of components and their relationships

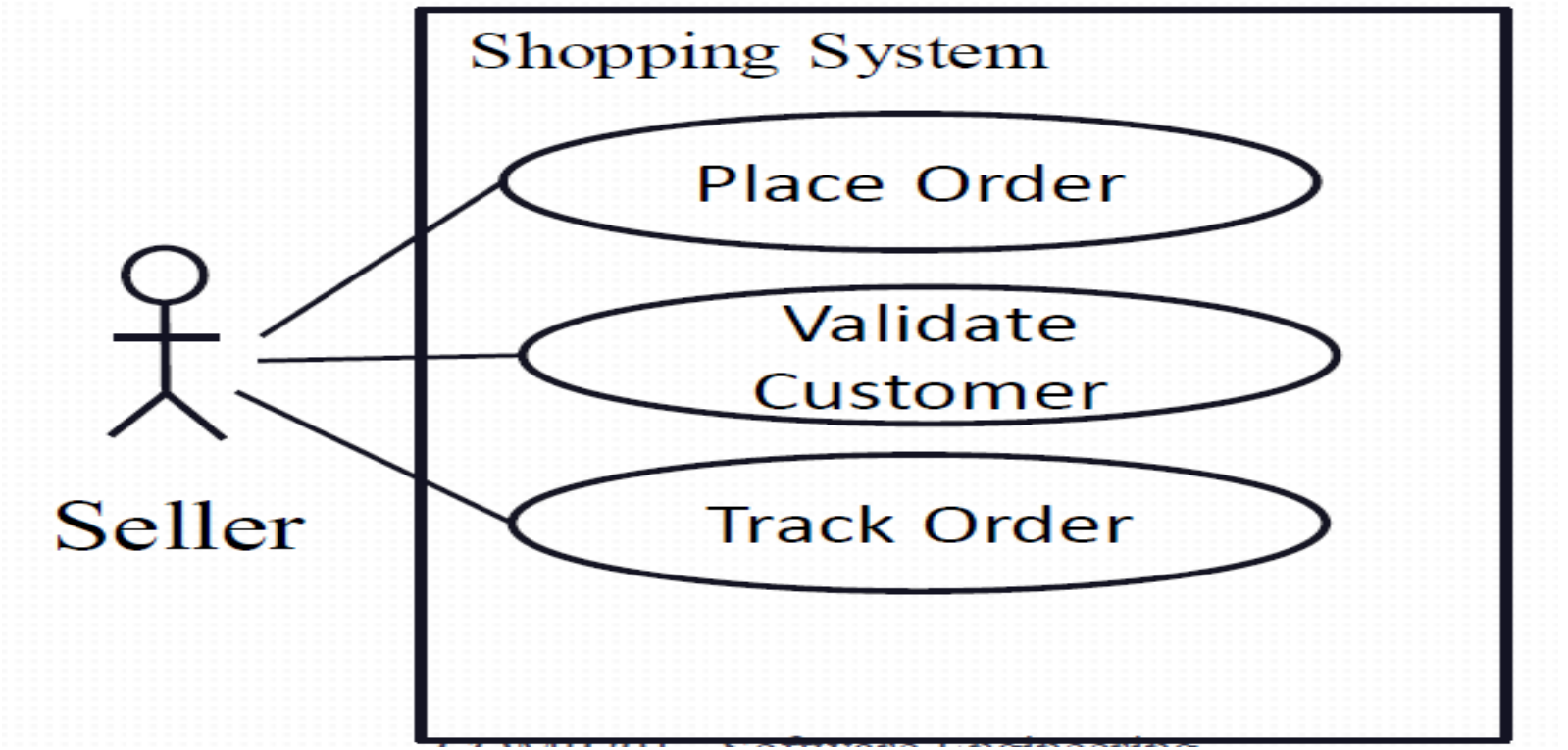


d)deployment diagram –graphically represents a set of processing nodes together with the components hosted on them and their relationships

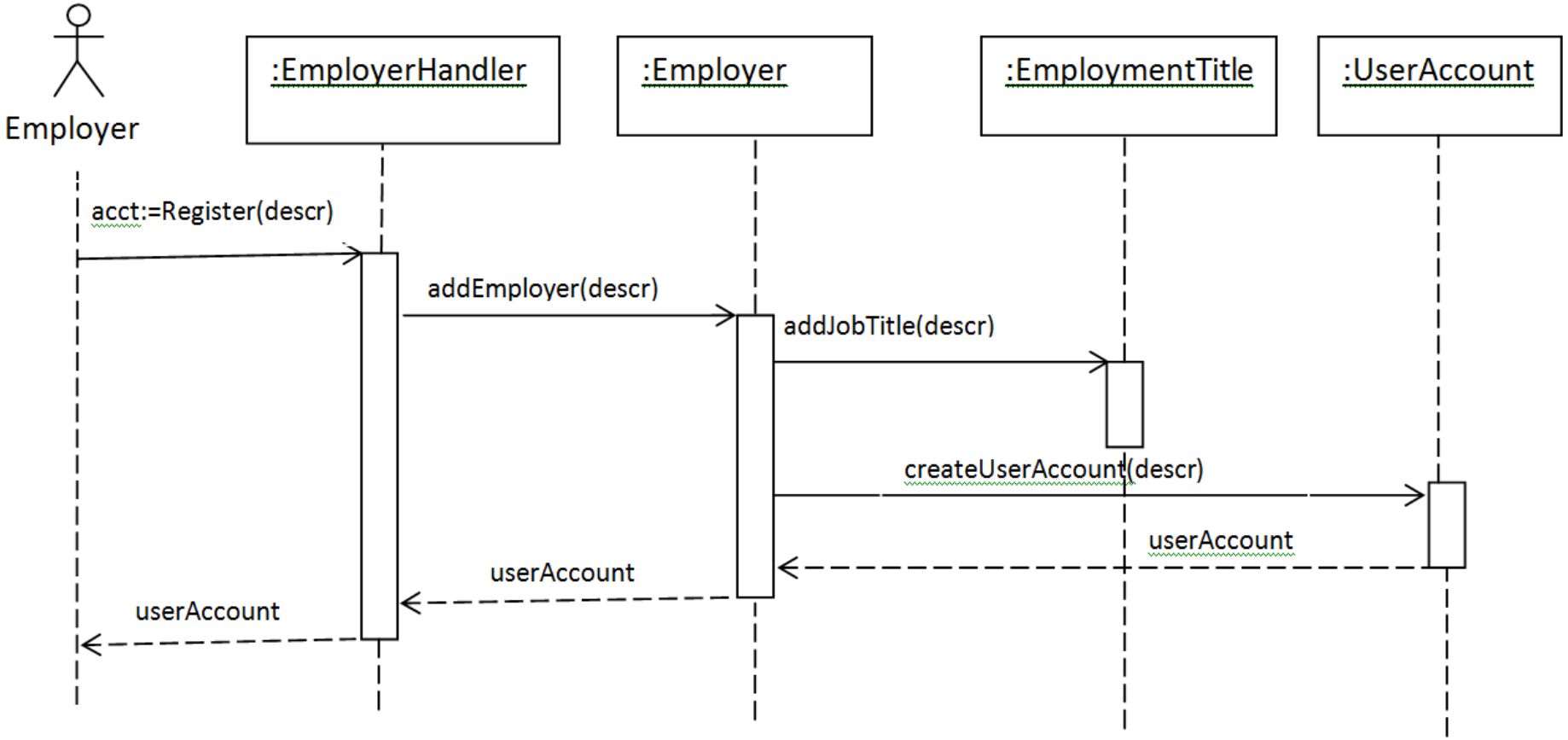


3.83dynamic modeling UML diagrams

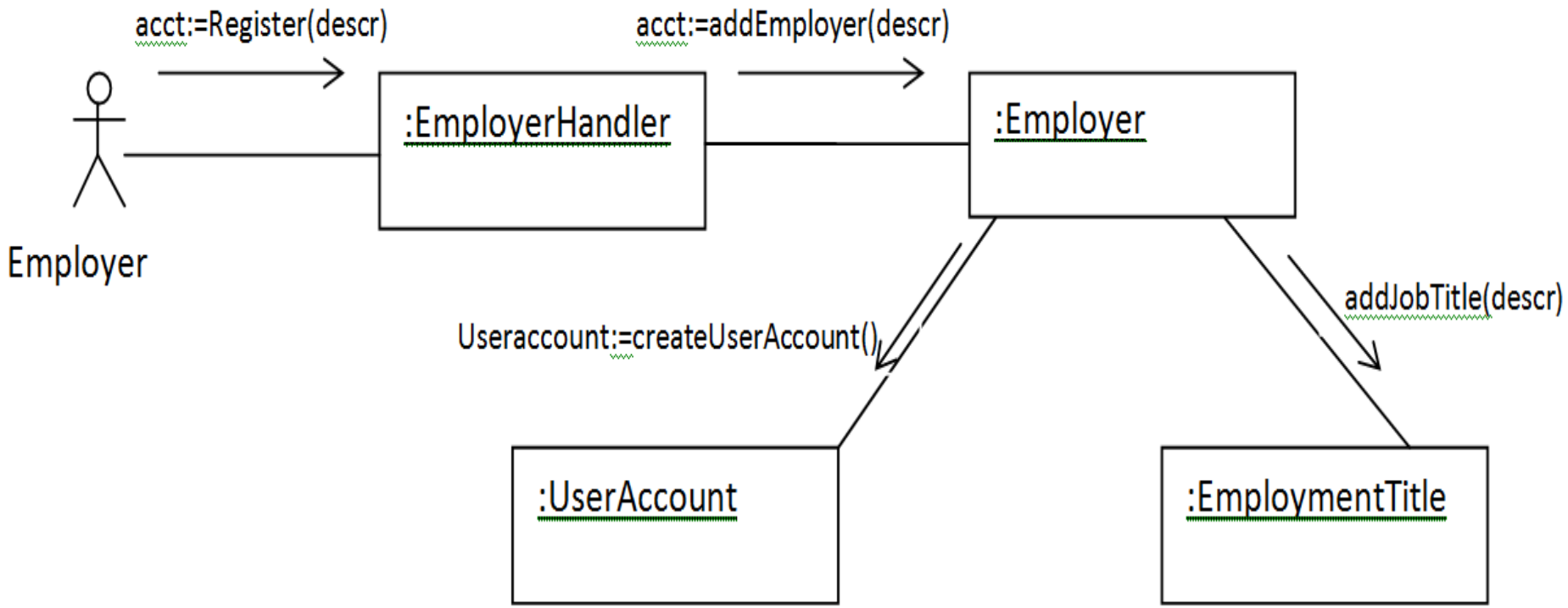
a)use case diagram –graphically represents a set of use cases(functionalities), actors(users) and their relationships



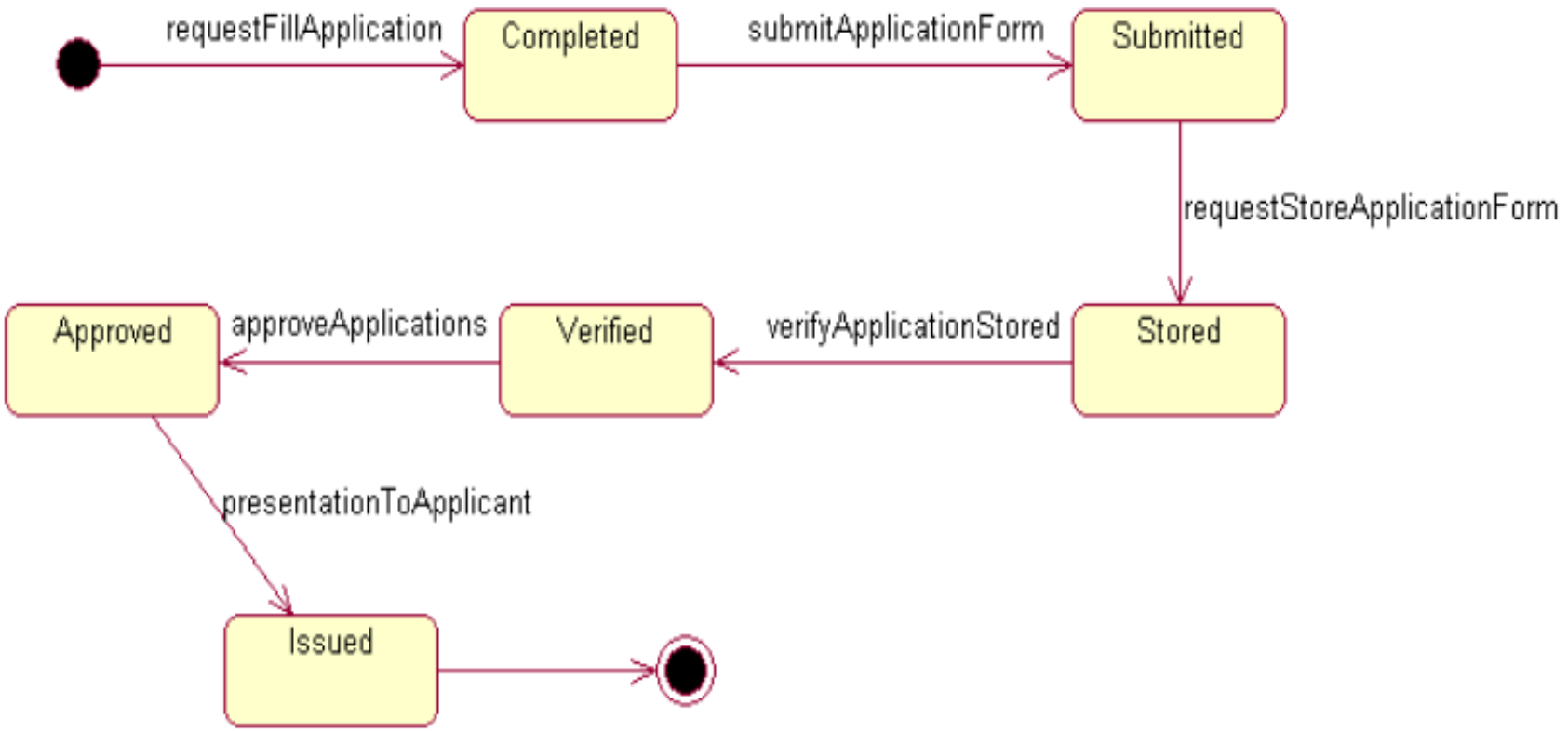
b)sequence diagram –graphically represents a set of interactions consisting of a set of objects and their messages with emphasis on the chronological ordering of messages



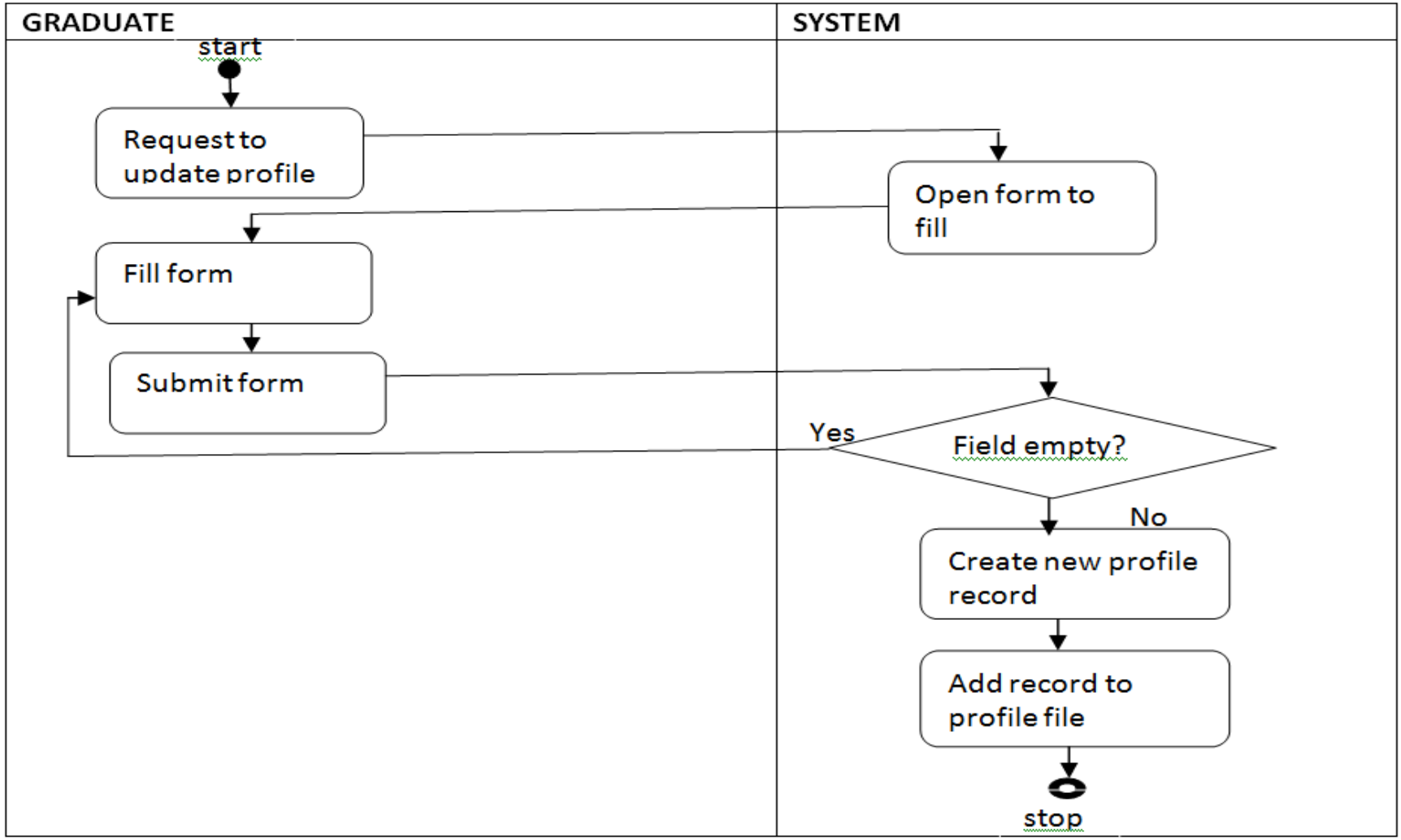
c)collaboration diagram –graphically represents the structural organization of objects that send and receive messages during interaction



d)statechart diagram –graphically represents the state machine of objects showing their states, transition, events and activities



e)activity diagram –graphically represents both the flow of control among objects and data flows from one activity to another



3.84 modeling UML views

–UML views – models of different perspectives of a system produced using specific set of diagrams

–essentially there are five UML modeling views:

1) Use Case View
2) Design View
3) Process View
4) Implementation View
5) Deployment View

1) Use Case View

–describes the behavior of the system as seen by its end users, analysts and testers.

–UML diagrams used to model this view are use case diagram and activity diagram

2) Design View

–describes structure of the system in terms of concepts that form solution to the underlying problem

–UML diagrams used to model this view are class diagram, object diagram, interaction diagrams, and state chart diagram

3) Process View

–describes the systems mechanisms for concurrency and synchronization of its behavior

–UML diagrams used to model this view are interaction diagrams and activity diagram

4) Implementation View

–describes the system in terms of components that are used to assemble it physically.

–UML diagram used to model the view is component diagram

5) Deployment View

–describes the system in terms of nodes that form hardware topology on which it will execute.

–UML diagram used to model the view is deployment diagram