Nadar Saraswathi College of Engineering & Technology

NSCET E-LEARNING PRESENTATION

LISTEN … LEARN… LEAD…

Nadar Saraswathi College of Engineering & Technology

COMPUTER SCIENCE AND ENGINEERING

III YEAR / V SEMESTER

CS8592 OBJECT ORIENTED ANALYSIS AND DESIGN

A.DURAIMURUGAN , M.E, (Ph.d)

ASSISTANT PROFESSOR

Nadar Saraswathi College of Engineering & Technology,

Vadapudupatti, Annanji (po), Theni – 625531.

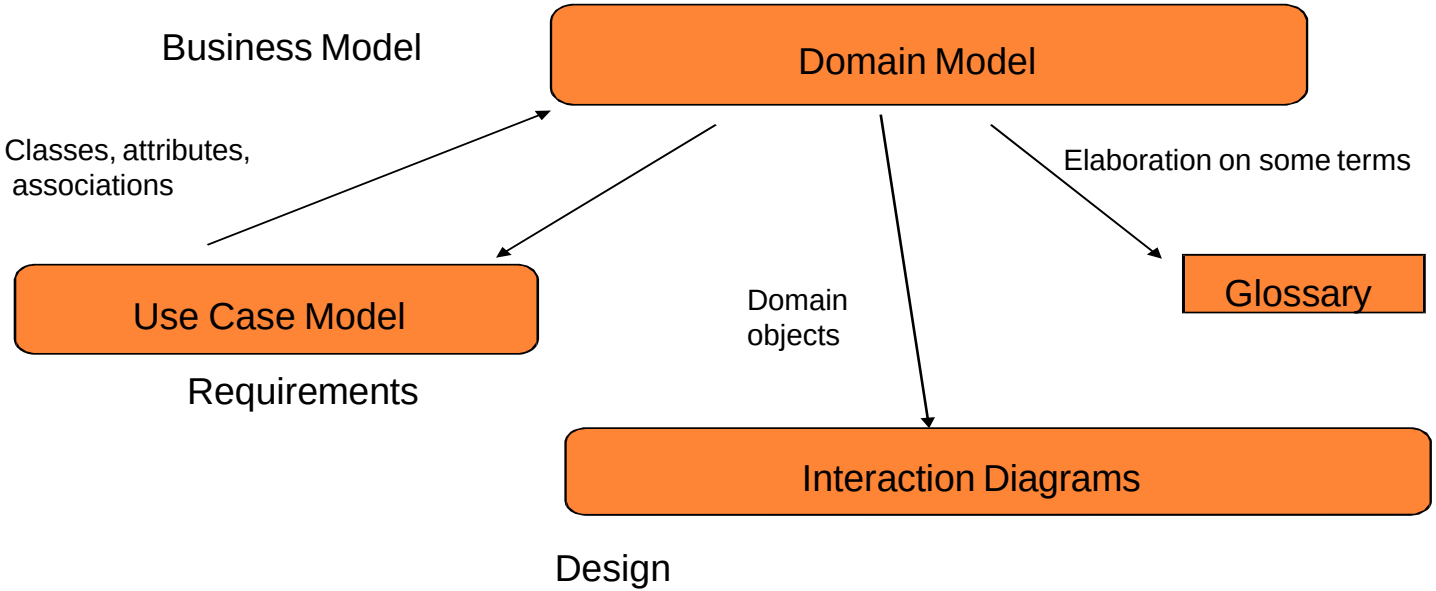
Nadar Saraswathi College of Engineering & Technology

UNIT II

STATIC UML DIAGRAMS

Nadar Saraswathi College of
Engineering & Technology

DOMAIN MODEL RELATIONSHIPS



WHAT IS A DOMAIN MODEL ?

A domain model:

- Illustrates meaningful conceptual classes in a problem domain.
- Is a representation of real-world concepts, not software components.
- Is NOT a set of diagrams describing software classes, or software objects and their responsibilities.

A DOMAIN MODEL IS THE MOST IMPORTANT OO ARTIFACT

- Its development entails identifying a rich set of conceptual classes, and is at the heart of object oriented analysis.
- It is a visual representation of the decomposition of a domain into individual conceptual classes or objects.
- It is a visual dictionary of noteworthy abstractions.

THINK OF CONCEPTUAL CLASSES IN TERMS OF:

- Symbols – words or images
- Intensions – its definition
- Extensions – the set of examples to which it applies
- Symbols and Intensions are the practical considerations when creating a domain model

DECOMPOSITION:

- A central distinction between Object-oriented analysis and structured analysis is the division by objects rather than by functions during decomposition.
- During each iteration, only objects in current scenarios are considered for addition to the domain model.

Interaction Diagram

From the name Interaction it is clear that the diagram is used to describe some type of interactions among the different elements in the model.

• Interaction diagrams are models that describe how a group of objects interact collaborate in some behavior - typically a single use-case.

• Purpose

– To capture dynamic behaviour of a system.

– To describe the message flow in the system.

– To describe structural organization of the objects. – To describe interaction among objects.

Drawing the interaction diagram

The purpose of interaction diagrams are to capture the dynamic aspect of a system.

• Dynamic aspect can be defined as the snap shot of the running system at particular moment.

• So the following things are to identified clearly before drawing the interaction diagram:

– Objects taking part in the interaction.

– Message flows among the objects. –

The sequence in which the messages are flowing.

– Object organization.

Sequence Diagram

Describe the flow of messages, events, actions between objects .

An important characteristic of a sequence diagram is that time passes from top to bottom and model important runtime interactions between the parts that make up the system.

Typically used to document and understand the logical flow of the system

Sequence Diagram Key Parts

- Participant: object or entity that acts in the diagram message: communication between participant objects
- The axes in a sequence diagram:

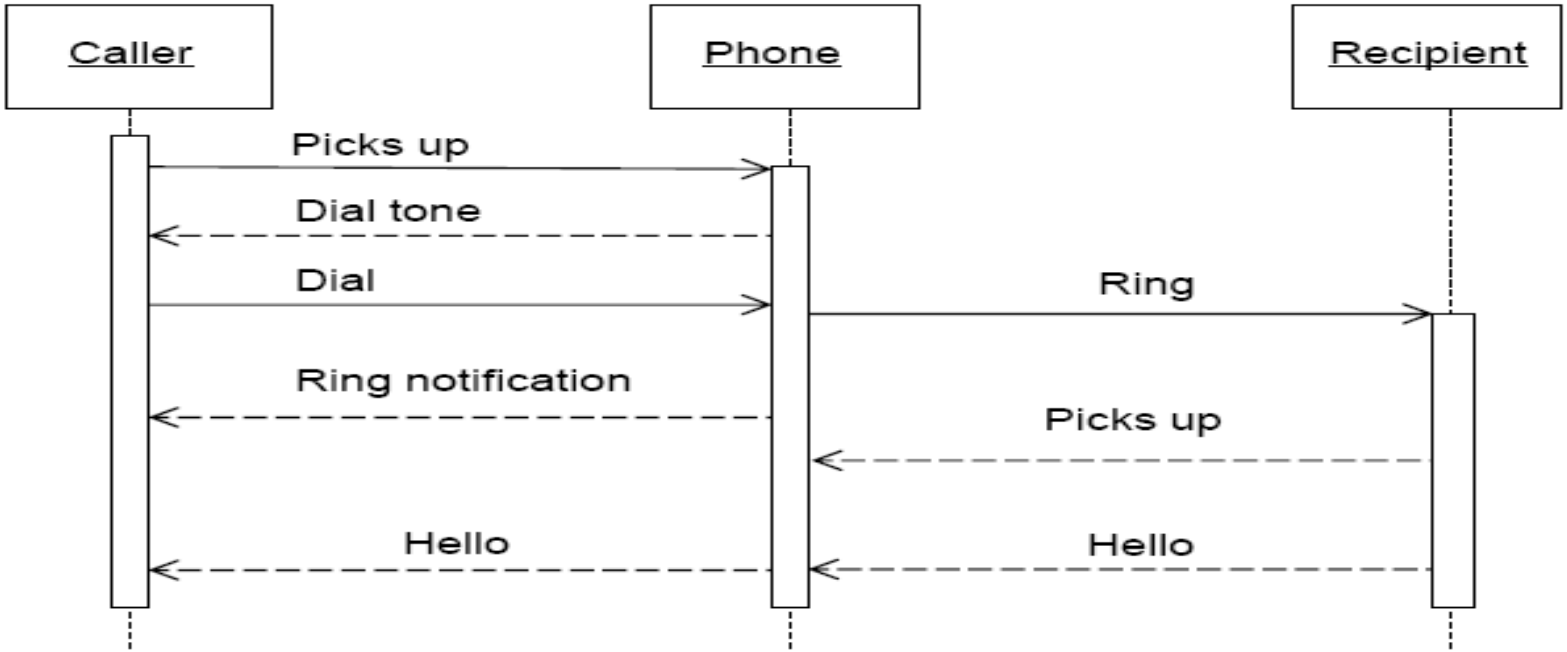
–horizontal: which object/participant is acting

–vertical: time (down -> forward in time)

Sequence Diagrams Dimensions

• Time. The vertical axis represents time proceedings (or progressing) down the page. Note that Time in a sequence diagram is all a about ordering, not duration. The vertical space in an interaction diagram is not relevant for the duration of the interaction.

• Objects. The horizontal axis shows the elements that are involved in the interaction. Conventionally, the objects involved in the operation are listed from left to right according to when they take part in the message sequence.

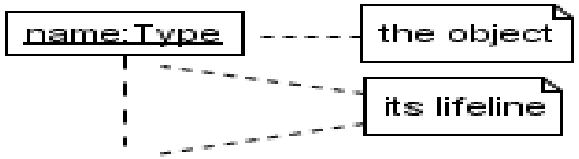


Object

- Objects as well as classes can be **targets on a**
- sequence diagram, which means that
- messages can be sent to them.
- A **target is displayed as a rectangle with some** text in it. Below the target, its lifeline extends for as long as the target exists. The lifeline is displayed as a vertical dashed line.

Object

- The basic notation for an object is
- where 'name' is the name of the object in the context of the diagram and 'Type' indicates the type of which the object is an instance.



- Both name and type are optional, but at least one of them should be present.

Lifelines in UML diagrams

- In UML diagrams, such as sequence or communication diagrams, lifelines represent the objects that participate in an interaction.
- Lifeline is a named element which represents an individual participant in the interaction.

For example, in a banking scenario, lifelines can represent objects such as a bank system or customer.

- Each instance in an interaction is represented by a lifeline.

- Lifeline in a sequence diagram is displayed with its name and type in a rectangle, which is called the head.
- The head is located on top of a vertical dashed line, called the sytem, which represents the timeline for the instance of the object.

Messages

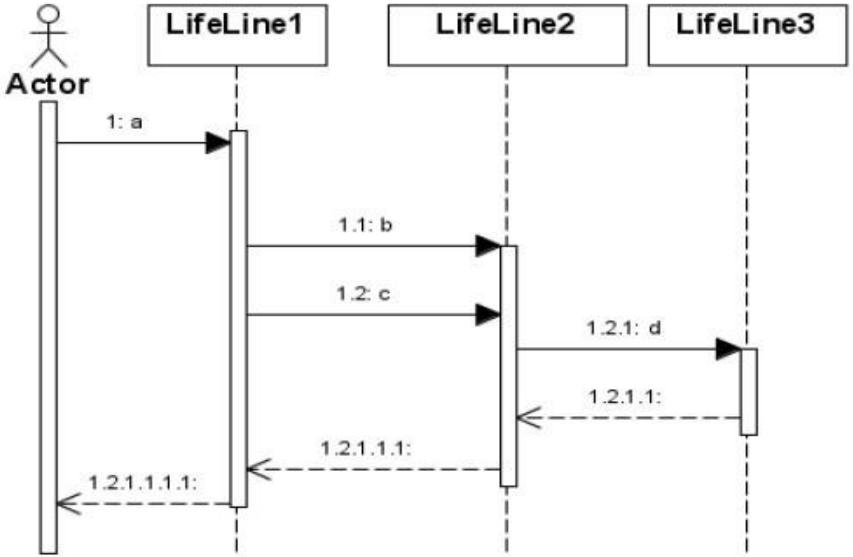
- Messages (or signals) on a sequence diagram are specified using an arrow from the participant (message caller) that wants to pass the message to the participant (message receiver) that is to receive the message.
- A Message (or stimulus) is represented as an arrow going from the sender to the top of the focus of control (i.e., execution occurrence) of the message on the receiver's lifeline.
- Near the arrow, the name and parameters of the message are shown.

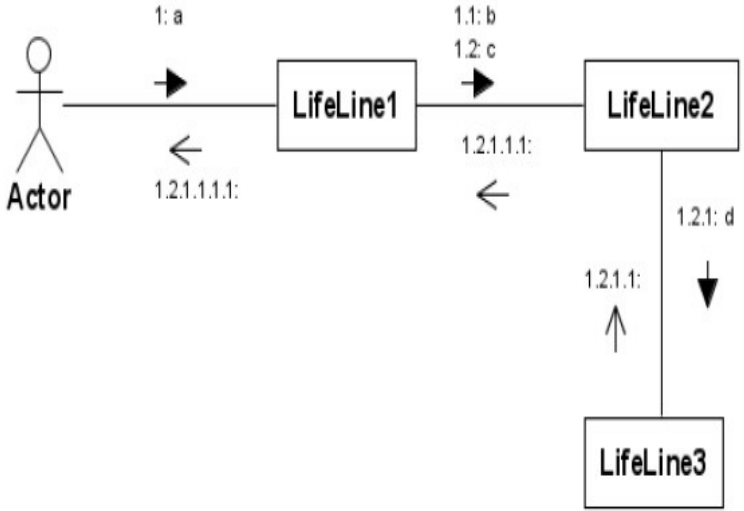
Communication diagram

- A communication diagram, is an interactiondiagram that shows similar information to sequence diagrams but its primary focus is on object relationships.
- Formerly known as collaboration diagram.
- Shows interactions between objects and/or parts (represented as lifelines) using sequenced messages in a free-form arrangement.

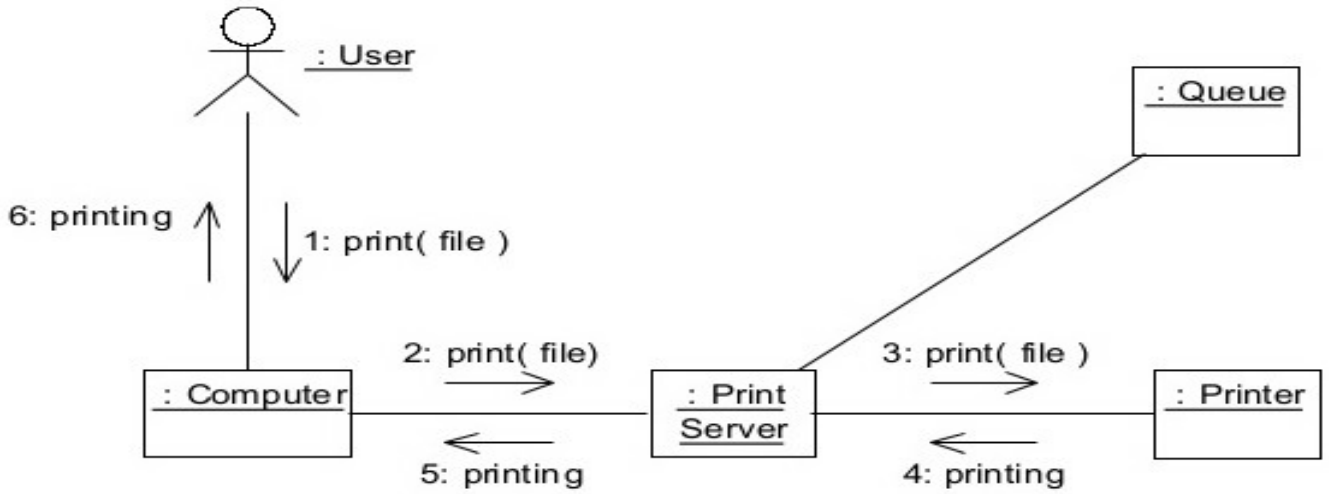
Communication diagram

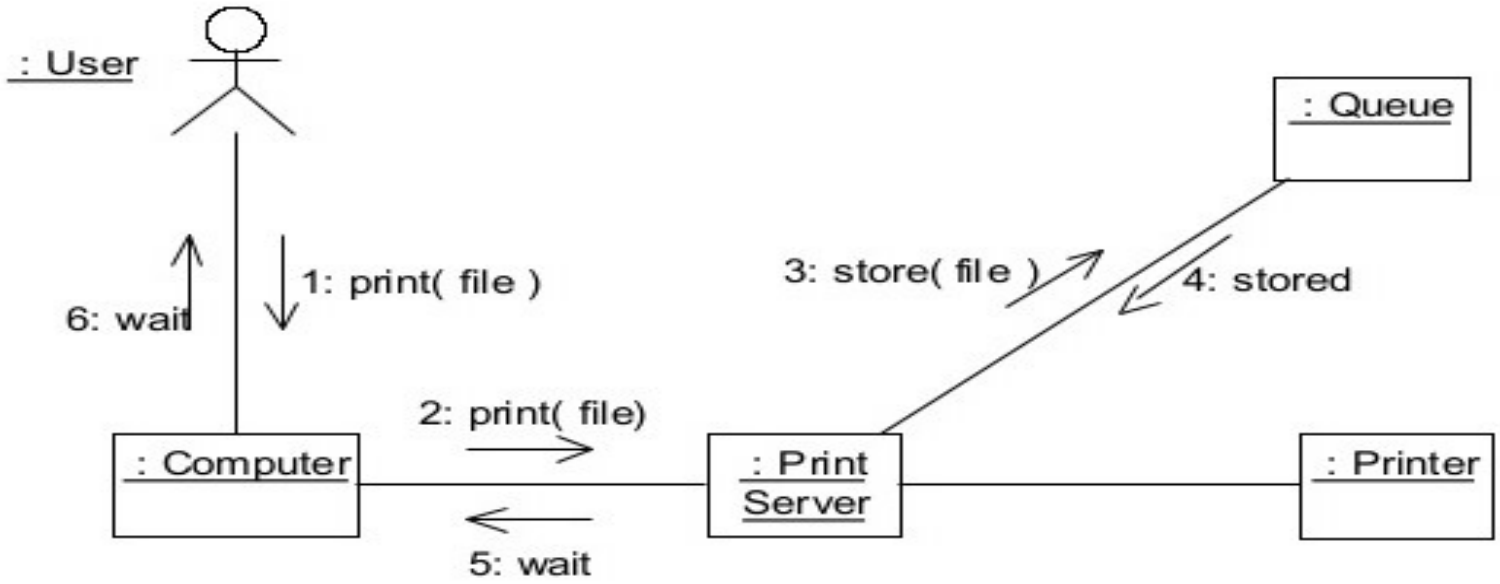
- Model collaborations between objects or roles that deliver the functionalities of use cases and operations.
- Capture interactions that show the passed messages between objects and roles within the collaboration.
- Support the identification of objects (hence classes) that participate in use cases.





Communication diagram





ASSOCIATIONS

OBJECTIVES

- Identify associations for a domain model.
- Distinguish between need-to-know and
- comprehension-only associations.

INTRODUCTION

- Identify associations of conceptual classes needed to satisfy the information requirements of current scenarios.
- Also identify the aid in comprehending the domain model.

ASSOCIATIONS

- An association is a relationship between instances of types that indicates some meaningful and interesting connection.

USEFUL ASSOCIATIONS

- Associations for which knowledge of the relationship needs to be preserved for some duration. Associations derived from the Common Associations List.

UML ASSOCIATION NOTATION

- An association is represented as a line between classes with an association name.
- Associations are inherently bidirectional.
- Optional reading direction arrow is only an aid to the reader of the diagram.

FINDING ASSOCIATIONS

- COMMON ASSOCIATIONS LIST
- The common categories that are worth considering are:
- A is a physical part of B . Eg: Wing-Airplane

ATTRIBUTES

- After establishing classes based on the concepts of use case scenarios, the scenarios are examined to discover attributes.
- Attributes are logical data values of an object.

VALID ATTRIBUTE TYPES

- Keep attributes simple
- Distinguish between conceptual and implementation perspectives
- Identify data types

- NON PRIMITIVE DATA TYPE (1)
- Represent what may be considered a primitive data type (such as a number or string) as a non primitive class if:
- It is composed of separate sections. phone number, name of person
- There are operations usually associated with it, such as parsing or validation. social security number
- It has other attributes promotional price could have a start date and end date

NON PRIMITIVE DATA TYPE (2)

- It has a quantity with a unit. payment amount has a unit of currency
- It has abstraction of one or more types with some of these qualities.
- item identifier in the sales domain is a generalization of types such as Universal product code(UPC) or European Article Number(EAN)

NON PRIMITIVE DATA TYPE (3)

- Applying these guidelines to the POS domain model yields the following analysis:
- The item identifier is an abstraction of various common coding codes schemes, including UPC-A, UPC-E, and the family of EAN schemes. These numeric coding schemes have subparts identifying the manufacturer, product and EAN

NON PRIMITIVE DATA TYPE (3)

- The price and the amount attribute should be non primitive Quantity or Money classes because they are quantities in a unit of currency
- The address attribute should be a non primitive Address class because it has separate sections

- DOMAIN MODEL CONCLUSION
- A relatively useful model has been created for the domain of the POS application.
- A good domain model captures the essential abstractions and information required to understand the domain in context of current requirements, and aids people in understanding the domain – its concepts , terminology, and the relationships.