

OBJECT ORIENTED ANALYSIS AND DESIGN

PGDCA -204

BLOCK 1: OBJECT ORIENTED MODELLING

**Dr. Babasaheb Ambedkar Open University
Ahmedabad**



OBJECT ORIENTED ANALYSIS AND DESIGN



Knowledge Management and
Research Organization
Pune



Editorial Panel

Author

PROF. K J Sharma

Language Editor

Mr. Vipul Shelke

Graphic and Creative Panel

Ms. K. Jamdal

Ms. Lata Dawange

Mr. Prashant Tikone

Copyright © 2015 Knowledge Management and Research Organization.

All rights reserved. No part of this book may be reproduced, transmitted or utilized in any form or by means of, electronic or mechanical, including photocopying, recording or by any information storage or retrieval system without written permission from us.

Acknowledgment

Every attempt has been made to trace the copyright holders of material reproduced in this book. Should an infringement have occurred, we apologize for the same and will be pleased to make necessary correction/amendment in future edition of this book.

The content is developed by taking reference of online and print publications that are mentioned in Bibliography. The content developed represents the breadth of research excellence in this multidisciplinary academic field. Some of the information, illustrations and examples are taken "as is" and as available in the references mentioned in Bibliography for academic purpose and better understanding by learner.'



ROLE OF SELF INSTRUCTIONAL MATERIAL IN DISTANCE LEARNING

The need to plan effective instruction is imperative for a successful distance teaching repertoire. This is due to the fact that the instructional designer, the tutor, the author (s) and the student are often separated by distance and may never meet in person. This is an increasingly common scenario in distance education instruction. As much as possible, teaching by distance should stimulate the student's intellectual involvement and contain all the necessary learning instructional activities that are capable of guiding the student through the course objectives. Therefore, the course / self-instructional material are completely equipped with everything that the syllabus prescribes.

To ensure effective instruction, a number of instructional design ideas are used and these help students to acquire knowledge, intellectual skills, motor skills and necessary attitudinal changes. In this respect, students' assessment and course evaluation are incorporated in the text.

The nature of instructional activities used in distance education self-instructional materials depends on the domain of learning that they reinforce in the text, that is, the cognitive, psychomotor and affective. These are further interpreted in the acquisition of knowledge, intellectual skills and motor skills. Students may be encouraged to gain, apply and communicate (orally or in writing) the knowledge acquired. Intellectual-skills objectives may be met by designing instructions that make use of students' prior knowledge and experiences in the discourse as the foundation on which newly acquired knowledge is built.

The provision of exercises in the form of assignments, projects and tutorial feedback is necessary. Instructional activities that teach motor skills need to be graphically demonstrated and the correct practices provided during tutorials. Instructional activities for inculcating change in attitude and behavior should create interest and demonstrate need and benefits gained by adopting the required change. Information on the adoption and procedures for practice of new attitudes may then be introduced.

Teaching and learning at a distance eliminates interactive communication cues, such as pauses, intonation and gestures, associated with the face-to-face method of teaching. This is particularly so with the exclusive use of print media. Instructional activities built into the instructional repertoire provide this missing interaction between the student and the teacher. Therefore, the use of instructional activities to affect better distance teaching is not optional, but mandatory.

Our team of successful writers and authors has tried to reduce this.

Divide and to bring this Self Instructional Material as the best teaching and communication tool. Instructional activities are varied in order to assess the different facets of the domains of learning.

Distance education teaching repertoire involves extensive use of self-instructional materials, be they print or otherwise. These materials are designed to achieve certain pre-determined learning outcomes, namely goals and objectives that are contained in an instructional plan. Since the teaching process is affected over a distance, there is need to ensure that students actively participate in their learning by performing specific tasks that help them to understand the relevant concepts. Therefore, a set of exercises is built into the teaching repertoire in order to link what students and tutors do in the framework of the course outline. These could be in the form of students' assignments, a research project or a science practical exercise. Examples of instructional activities in distance education are too numerous to list. Instructional activities, when used in this context, help to motivate students, guide and measure students' performance (continuous assessment)



PREFACE

We have put in lots of hard work to make this book as user-friendly as possible, but we have not sacrificed quality. Experts were involved in preparing the materials. However, concepts are explained in easy language for you. We have included many tables and examples for easy understanding.

We sincerely hope this book will help you in every way you expect.

All the best for your studies from our team!



OBJECT ORIENTED ANALYSIS AND DESIGN

Contents

BLOCK 1: OBJECT ORIENTED MODELLING

UNIT 1 INTRODUCTION TO OBJECT ORIENTED MODELLING

Object Oriented Modeling, Characteristics Object Oriented Modeling: Class and Objects, Links and Association, Generalization and Inheritance, An Object Model. Benefits of OO Modeling, Introduction to OOAD tools,

UNIT 2 ADVANCE MODELING CONCEPTS

Aggregation, Abstract Class, Multiple Inheritance, Generalization as an Extension, Generalization as a Restriction, Metadata, Constraints, An Object Model

BLOCK 2: OBJECT ORIENTED ANALYSIS AND SYSTEM DESIGN

UNIT 1 OBJECT ORIENTED ANALYSIS

Object Oriented Analysis, Problem Statement: an Example, Differences between Structured Analysis and Object Oriented Analysis, Analysis Techniques: Object Modeling, Dynamic Modeling, and Functional Modeling. Adding Operations. Analysis Iteration

UNIT 2 SYSTEM DESIGN

System Design: An Object Oriented Approach, Breaking into Subsystems, Concurrency Identification, Management of data store, Controlling events between Objects, Handling Boundary Conditions



BLOCK 3: OBJECT DESIGN AND DYNAMIC MODELING

UNIT 1 OBJECT DESIGN

Object Design for Processing, Object Design Steps, Designing a Solution, Choosing Algorithms, Choosing Data Structures, Defining Classes and delegation of Responsibilities to Methods

UNIT 2 DYNAMIC MODELING

Events, State and State Diagram, Elements of State Diagrams, Examples of State Diagrams, Advance Concepts in Dynamic Modeling, Concurrency, A Dynamic model

BLOCK 4: FUNCTIONAL MODELING AND UML

UNIT 1 FUNCTIONAL MODELING

Functional Models, Data Flow Diagrams, Features of a DFD, Design flaws in DFD, A Functional model, Relationship between Object, Dynamic, and Functional Models

UNIT 2 USING UML

UML: Introduction, Object Model Notations: Basic Concepts, Structural Diagrams: Class, Object, Composite, Package, Component, Deployment. Behavioral Diagrams: Use Case, Communication, Sequence, Interaction Overview, Activity, State. Modeling with Objects

UNIT 3 CASE STUDY

This unit will cover all the OOAD aspects Covered in previous units of this course.



Dr. Babasaheb
Ambedkar
Open University

PGDCA-204

OBJECT ORIENTED ANALYSIS AND DESIGN

BLOCK 1: OBJECT ORIENTED MODELLING

UNIT 1

INTRODUCTION TO OBJECT ORIENTED MODELLING 03

UNIT 2

ADVANCE MODELING CONCEPTS 17

Block Structure

Unit 1: Introduction to Object Oriented Modelling

Unit 2: Advance Modelling Concepts

UNIT 1: INTRODUCTION TO OBJECT ORIENTED MODELLING

Unit Structure

- 1.0 Learning Objectives**
- 1.1 Introduction**
- 1.2 Object Oriented Modelling**
- 1.3 Characteristics Object Oriented Modelling: Class and Objects**
- 1.4 Links and Association**
- 1.5 Generalization and Inheritance**
- 1.6 An Object Model**
- 1.7 Benefits of OO Modelling**
- 1.8 Introduction to OOAD tools**
- 1.9 Let Us Sum Up**
- 1.10 Answers for Check Your Progress**
- 1.11 Glossary**
- 1.12 Assignment**
- 1.13 Activities**
- 1.14 Case Study**
- 1.15 Further Readings**

1.0 Learning Objectives

After learning this unit, you will be able to understand:

- The basics of object oriented Modeling
- Define Objects and Classes
- Explain the concepts of links and Associations
- Explain the concept of Generalization and Inheritance
- Describe benefits of Object Oriented Modeling

1.1 Introduction

Object-oriented technology is incredibly wide and extensive. The users of computer systems found the effects of such technology as increasingly easy-to-use software applications and operating systems that are flexible that's catered by many industries like banking, telecommunications and television. Just in case of software technology, such object-oriented technology can cover object-oriented management of projects, computer hardware and computer aided software engineering, among others.

An interface is mixture of abstract methods. A class implements an interface, and can inherit abstract ways of interface. It's not a class, however if we tend to write it, it's equivalent to writing a class. Any it's seen that a class describes itself as an attributes and behaviours of an object. An interface contains behaviours that a class implements.

1.2 Object Oriented Modelling

It is one of the most objectives of the software engineering discipline to support the complicated and thus error-prone software development task by providing appropriate sophisticated concepts, languages, techniques, and tools to any or all stakeholders involved. A crucial and nowadays commonly accepted approach within software engineering is that the usage of a software development process model, where above all the software development task is separated into a series of dedicated subtasks. a substantial constituent of such a stepwise approach is the development of a system model. Such a model describes the requirements for the software system to be realized and forms an abstraction.

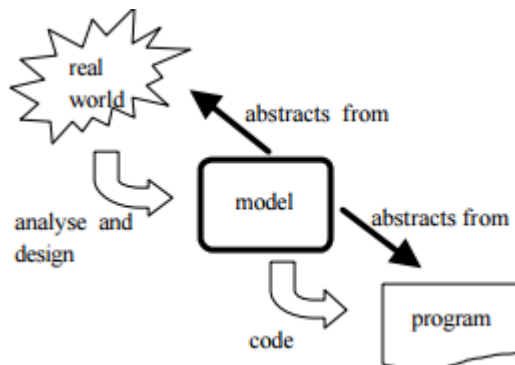


Fig 1.1 Model Representations

The success of object-oriented modelling approaches was hindered within the beginning of the 90ies due to the very fact that surely over 50 object-oriented

modelling approaches claimed to be the proper one. This so-called technique war came to an end through an industrial initiative, that pushed the development of the meanwhile industrially standardized object-oriented modelling language UML.

Check your progress 1

1. What is object oriented modeling strategy?
 - a. Use of algorithms
 - b. Use of objects
 - c. Use of classes
 - d. Both B and C

1.3 Characteristics Object Oriented Modelling: Class and Objects

Object oriented modelling is entirely a brand new way of thinking about problems. This methodology is all concerning visualizing the items by using models organized around real world ideas. Object oriented models help in understanding issues, communicating with experts from a distance, modelling enterprises, and designing programs and database. In object oriented modelling objects and their characteristics are described. In any system, objects come into existence for playing some role. In the method of defining the roles of objects, some options of object orientation are used.

Class and Objects

A class is a collection of things, or ideas that have the same characteristics. Each of these things or concepts is named as object. Classes define the basic words of the system being modelled. Using a set of classes as the core vocabulary of a software project tends to greatly facilitate understanding and agreement concerning the meanings of terms, and other characteristics of the objects in the system. Classes can serve as the foundation for data modelling. In OOM, the term classes is sometimes the base from that visual modelling tools—such as Rational Rose XDE, Visual Paradigm function and design the model of systems.

A class could be a pattern, template, or blueprint for a category of structurally identical items. The items created using the class are known as instances. This is

often referred to as the "class or `cookie cutter'" view. As you might guess, the instances are the "cookies."

A class could be a thing that consists of both a pattern and a mechanism for creating items supported that pattern. This is the "class as an `instance factory'" view; instances are the individual items that are "manufactured" using the class's creation mechanism.

A class is the set of all items created using a specific pattern. In another way, the class is that the set of all instances of that pattern.

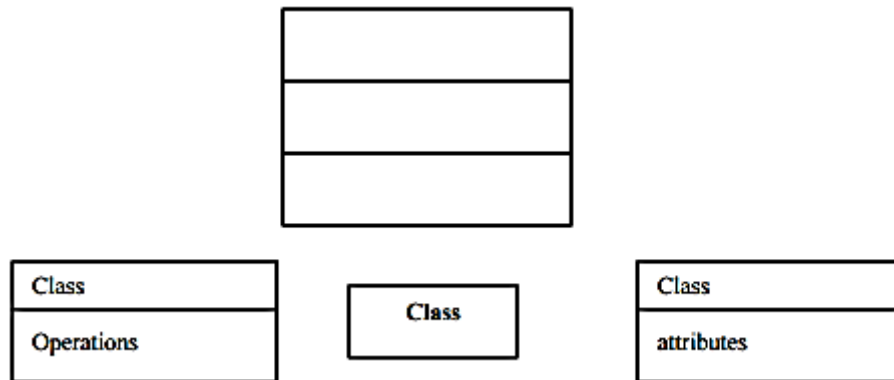


Fig 1.2 Class Notations

Objects

Objects are physical as they are present in universe around us. It is found that hardware; software, documents and concepts are all object samples. For use of modelling, an officer views staff, buildings, divisions, documents and benefits packages as related objects. An automobile person would see tires, doors, engines, speed and fuel level in terms of objects, while atoms, molecules, volumes, and temperatures are objects according to chemist which can be thought of as making an object oriented simulation of chemical reaction.

Normally it is viewed that objects can be considered as particular state. The state of an object is basically the condition of an object, or set of circumstances describing the item. It is generally seen that that people are talking about state information which in particular relates to specific object. It is noticed that state of bank account object will cover latest and available balance, which shows state of clock object that is available at required time showing state of lightweight bulb which could be placed at on or off state. For complex objects like a human being or an automobile, an entire description of the state may be very complex. Fortunately, when we use objects to model real world or imagined things, we have

a tendency to generally restrict the possible states of the objects to only people who are relevant to our models.

Check your progress 2

1. What is class?
 - a. Collection of similar objects.
 - b. Collection of different types of objects
 - c. Group of information
 - d. None of these

1.4 Links and Association

Links and association are modes of creating link which exists among objects and classes. It is noticed that both links and association bears similar feature whereas links establishes among objects while association establishes among class.

Links: In object modelling links provides a relationship between the objects. These objects or instance may be same or different in data structure and behaviour. Thus a link is a physical or conceptual connection between instances (or objects).

Associations: the object modelling describes as a group of links with common structure and customary semantics. All the links among the object are the varieties of association among the same classes. The association is that the relationship among classes.

1. Association
2. Association with inverse direction
3. Association between student and university

Degree of association is:

1. Unary association (degree of one)
2. Binary Association (degree of two)
3. Ternary Association (degree of three)

4. Quaternary Association (degree of four)
5. Higher order association (more than four)

Check your progress 3

1. What is meant by link in object modeling?
 - a. Way to describe association
 - b. Physical or conceptual connection between instances
 - c. Relationship between classes
 - d. None of these

1.5 Generalization and Inheritance

Generalization and inheritance are powerful abstractions for sharing the structure and/or behaviour of one or a lot of classes. Generalization is that the relationship between a class and it defines a hierarchy of abstraction in which subclasses (one or more) inherit from one or more super classes. The notation for generalization is a triangle connecting a super class to its subclasses. The super class is connected by a line to the top of the triangle. The subclasses are connected by lines to a horizontal bar attached to the base of the triangle.

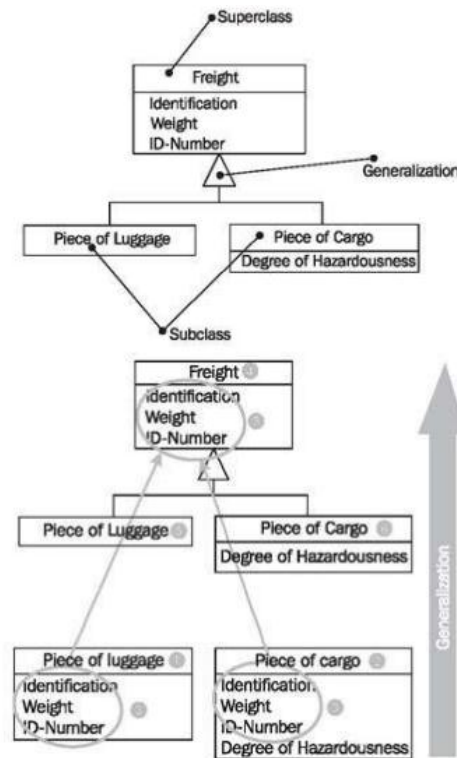


Fig 1.3 Generalisation Concepts

Inheritance is taken within the sense of code reuse within the object oriented development. Throughout modelling, we look at the resulting classes, and try to group similar classes along so code utilize can be enforced. Generalization, specialization, and inheritance have terribly close association. Generalization is used to refer to the relationship among classes, and inheritance is employed for sharing attributes and operations using the generalization relationship.

Inheritance and generalisation have in common that they both have the ability to define an abstract "contract" or "protocol" that a group of concrete classes following this "contract" have to implement. this gives you the ability to handle a number of comparable classes in an exceedingly uniform way, that permits you to write down additional compact, abstract code, that once more permits code reuse. Consider a class implements interface example where one interface extends another interface as:

```
interface Printable{
void print();
}
interface Showable extends Printable{
void show();
```

```
}  
class Testinterface2 implements Showable  
{  
public void print(){System.out.println("Sanjay");}  
public void show(){System.out.println("Mathur");}  
public static void main(String args[]){  
Testinterface2 obj = new Testinterface2();  
obj.print();  
obj.show();  
}  
}
```

If we run this program we get an output as:

Sanjay

Mathur

Check your progress 4

1. What is inheritance?
 - a. Code reuse within object modeling
 - b. Getting properties of one class into another class
 - c. It is meant for sharing attributes.
 - d. All of these

1.6 An Object Model

The object model visualizes all the elements in a software application in terms of objects. The example of Object Model is shown in fig 1.3

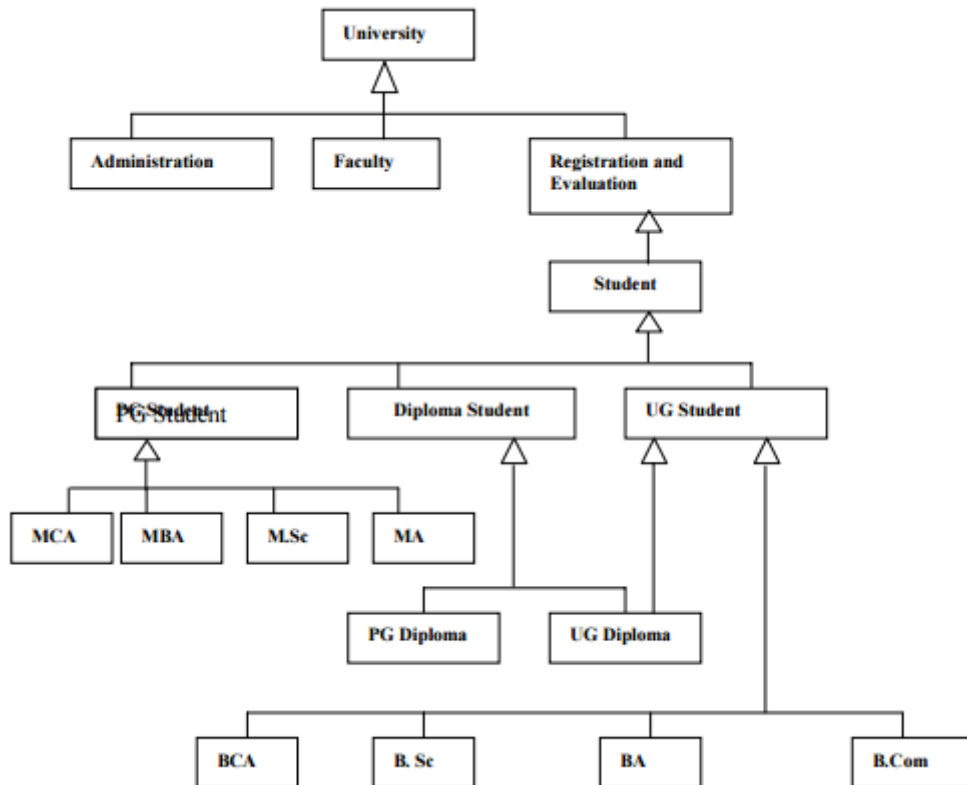


Fig 1.3 Object Model

- An object contains values stored in instance variables within the object.
- Unlike the record-oriented models, these values are themselves objects.
- Thus objects contain objects to an arbitrarily deep level of nesting.
- An object also contains bodies of code that operate on the the object.
- These bodies of code are called methods.
- Objects that contain the same types of values and the same methods are grouped into classes.
- A class may be viewed as a type definition for objects.
- Analogy: the programming language concept of an abstract data type.
- The only way in which one object can access the data of another object is by invoking the method of that other object.
- This is called sending a message to the object.

Check your progress 5

1. What is meant by sending message to object?
 - a. Passing parameters to object
 - b. Invoking the method of another object
 - c. Passing one object definition to another
 - d. None of these

1.7 Benefits of OO Modelling

Here are a number of the advantages of the object-oriented approach:

Reduced Maintenance: the first goal of object-oriented development is that the assurance that the system can enjoy a longer life while having far smaller maintenance costs. As a result of most of the processes among the system is encapsulated, the behaviours is also reused and incorporated into new behaviours.

Real-World Modelling: Object-oriented system tends to model the real world in a lot of complete fashion than do traditional methods. Objects are organized into classes of objects, and objects are related to behaviours. The model relies on objects, rather than on data and processing.

Improved reliability and Flexibility: Object-oriented system promise to be far more reliable than traditional systems, primarily because new behaviours is "built" from existing objects. Because objects is dynamically called and accessed, new objects could also be created at any time. The new objects might inherit data attributes from one, or several other objects. Behaviours are also inherited from super-classes, and novel behaviours are also added without effecting existing systems functions.

High Code Reusability: once a new object is created, it'll automatically inherit the data attributes and characteristics of the class from that it absolutely were spawned. The new object will inherit the data and behaviours from all super classes in which it participates. Once a user creates a new type of a widget, the new object behaves "wigitty", while having new behaviours that are defined to the system.

Check your progress 6

1. What are the benefits of object oriented modeling?
 - a. Improved reliability
 - b. Improved flexibility
 - c. Reduced maintenance
 - d. All of these

1.8 Introduction to OOAD tools

The major phases of software development using object-oriented methodology are object-oriented analysis, object-oriented design, and object-oriented implementation. Object-Oriented Analysis is a stage where the problem is formulated, user requirements are identified, and model is created on real-world objects. The analysis produces models on how the desired system should function and how it must be developed. The models do not include any implementation details so that it can be understood and examined by any non-technical application expert. Object-Oriented Design includes two main stages, namely, system design and object design.

There are three main tools used in object-oriented analysis and design techniques:

- Class diagrams/templates.
- Object diagrams.
- Object state diagrams

Class diagrams are used to model key abstractions in the problem domain and their relationships with each other. Object diagrams are used to model the interactions between objects, whereas object state diagrams model the dynamic behaviour within a single object. An object state diagram shows all the possible states of an object and the allowed transition between the states.

Check your progress 7

1. What are the two main stages of object –oriented design?
 - a. System design
 - b. Object design
 - c. Both of these
 - d. None of these

1.9 Let Us Sum Up

In this unit we have learnt that object oriented technology is wide and extensive which increases easy-to-use software applications with operating systems that is flexible. It is seen that object oriented modelling is brand new way of thinking about problems in which you can visualize items by modelling that are arranged around real world ideas.

It is known that class consists of pattern and mechanism for creating items which supports pattern where class as an instance factory can be seen as an instance of individual items manufactured by class creation mechanism. We have studied that links and association are modes of creating link that exists in objects and classes with both links and association bears same feature and establishes link among objects and association establishes class.

The generalization and inheritance are strong abstractions for sharing structure and behaviour of classes and bears relationship among class and show branch of abstraction where subclasses inherit from super classes. The object oriented analysis is particular stage where problem gets framed with identification of user requirements and creation of modelling in real world objects.

1.10 Answers for Check Your Progress

Check your progress 1

Answers: (1 –d)

Check your progress 2

Answers: (1 –a)

Check your progress 3

Answers: (1 –b)

Check your progress 4

Answers: (1 –d)

Check your progress 5

Answers: (1 –b)

Check your progress 6

Answers: (1 –d)

Check your progress 7

Answers: (1 –c)

1.11 Glossary

1. **Package** - It is a collection of types that gives access protection and name space management in Java.
2. **Interface** - In programming, interface is a mixture of abstract methods where class implements an interface.

1.12 Assignment

Explain the Object oriented modelling and its advantages.

1.13 Activities

Study OOAD tools.

1.14 Case Study

Study the difference between generalisation and specialisation and also justify how they are useful in Object oriented modelling

1.15 Further Readings

1. Learning Programming by Peter Norvig's.
2. Approach programming with a more positive by P.Brian.Mackey.

UNIT 2: ADVANCE MODELING CONCEPTS

Unit Structure

- 2.0 Learning Objectives**
- 2.1 Introduction**
- 2.2 Aggregation**
- 2.3 Abstract Class**
- 2.4 Multiple Inheritance**
- 2.5 Generalization as an Extension**
- 2.6 Generalization as a Restriction**
- 2.7 Metadata**
- 2.8 Constraints**
- 2.9 An Object Model**
- 2.10 Let Us Sum Up**
- 2.11 Answers for Check Your Progress**
- 2.12 Glossary**
- 2.13 Assignment**
- 2.14 Activities**
- 2.15 Case Study**
- 2.16 Further Readings**

2.0 Learning Objectives

After learning this unit, you will be able to understand:

- Study about features of Object Model
- Study about advantages of Abstract Class
- Study about Multiple Inheritance
- Study about Extension and Restriction

2.1 Introduction

Inheritance are often defined as the process wherever one object acquires the properties of another. With the use of inheritance the information is made manageable in a hierarchical order. Once we talk about inheritance, the most unremarkably used keyword would be extends and implements. These words would determine whether one object IS-A style of another. By using these keywords we will create one object acquires the properties of another object. Inheritance may be a compile-time mechanism. A super-class will have any number of subclasses. However a subclass will have just one super class. This is because Java does not support multiple inheritances.

2.2 Aggregation

Aggregation is an extension of association mean aggregation is a strong form of association within which an aggregate object is formed of components. Components are a part of the aggregate. So aggregation is the “part-whole” or “a-part-of” relationship.

Aggregation is a special case of association, a directional association between objects. When an object ‘has-a’ another object, then you have got an aggregation between them. Direction between them specified that object contains the other object. Aggregation is additionally known as a “Has-a” relationship.

Check your progress 1

1. What is Aggregation?
 - a. IS-A relationship
 - b. a-part-of relationship
 - c. Has-a relationship
 - d. Both B and C

2.3 Abstract Class

In Java, a class is a group of objects having certain common properties. It is a sort of template or blueprint from where objects are created. So a class in java contains:

- Data member
- Method
- Constructor
- Block
- Class and interface

The syntax to declare a class is:

```
Class <class_name>{  
    Data member;  
    method  
}
```

Consider an example of Object and Class

In this example, a Student class is created having two data members' id and name. Now the object of Student class is created with the help of new keyword and printing objects value as:

```
class Student1 {  
    int id;//data member (also instance variable)  
    String name;//data member(also instance variable)  
  
    public static void main(String args[]){  
        Student1 s1=new Student1();//creating an object of Student  
        System.out.println(s1.id);  
        System.out.println(s1.name);  
    }  
}
```

If we run the above program, we find:

Output: 0 null

Abstract Class

An abstract class defines an abstract concept that can't be instantiated. we can't create object of abstract class, it will only be inherited. Abstract class

normally represents concept with general actions associated with it. Abstract class can't be instantiated; it will solely be inherited while interfaces should be implemented. Abstract class will have implemented methods which interfaces will have only definitions of the methods without implementation.

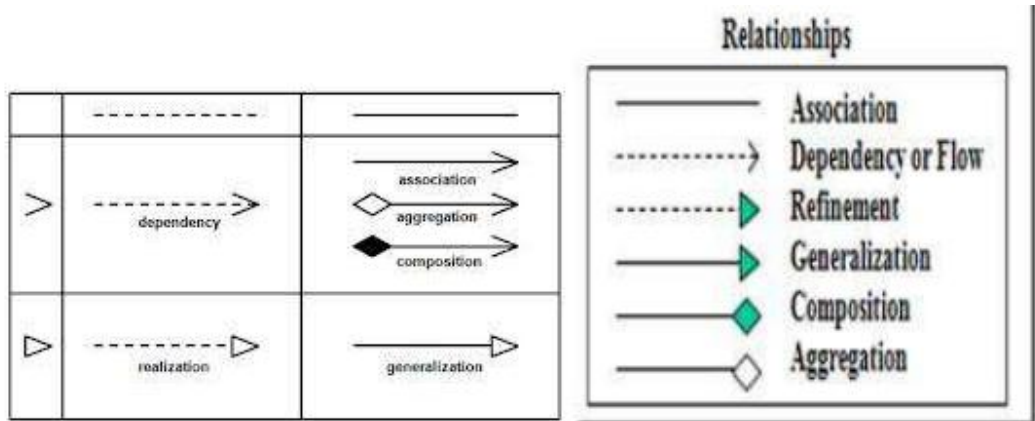


Fig 2.1 Abstract Class relationship

Check your progress 2

1. Abstract class are _____.

 - a. Inherited
 - b. Implemented
 - c. Instantiated
 - d. None of these

2.4 Multiple Inheritance

Inheritance is the most important features of Object oriented Programming that permits a {class|a category} to use properties and methods of another class. In this, the derived class is named as subclass and base class is named as super-class. It's seen that a derived class adds additional variables and methods which can differentiate derived class from base class. The syntax is shown below:

```
public class ChildClass extends BaseClass {
    // derived class methods extend and possibly override
}
```

Consider an example:

```
// A class to display the attributes of the vehicle
class Vehicle {
    String color;
    int speed;
    int size;
    void attributes() {
        System.out.println("Color : " + color);
        System.out.println("Speed : " + speed);
        System.out.println("Size : " + size);
    }
}

// A subclass which extends for vehicle
class Car extends Vehicle {
    int CC;
    int gears;
    void attributescar() {
        // The subclass refers to the members of the superclass
        System.out.println("Color of Car : " + color);
        System.out.println("Speed of Car : " + speed);
        System.out.println("Size of Car : " + size);
        System.out.println("CC of Car : " + CC);
        System.out.println("No of gears of Car : " + gears);
    }
}

public class Test {
    public static void main(String args[]) {
        Car b1 = new Car();
        b1.color = "Blue";
        b1.speed = 200 ;
        b1.size = 22;
        b1.CC = 1000;
        b1.gears = 5;
        b1.attributescar();
    }
}
```

If we run the above program, we get:

```
Color of Car : Blue
Speed of Car : 200
Size of Car : 22
CC of Car : 1000
No of gears of Car : 5
```

Inheritance results as an effective method which will share code among various classes having some traits in common by allowing classes to have different parts. Fig 2.2 called as Vehicle class which carries two subclasses as Car and Truck.

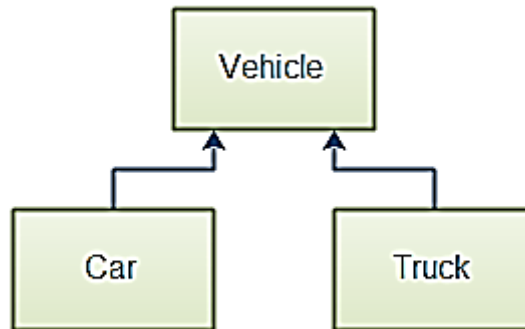


Fig 2.2 Vehicle class

It is seen that Vehicle class belongs to superclass of Car and Truck that are subclasses of Vehicle. Here, the Vehicle class contain those fields and methods which are Vehicles needed whereas Car and Truck have fields and methods particular to Cars and Trucks.

It is seen that many people declare that inheritance is a way to categorize particular class. It is studied that a Car is a Vehicle where a Truck is also a Vehicle. So, it is not how you determine super classes and subclasses in your application. It simply shows how you need to work with them.

As seen, when a subclass extends a super class, then all protected and public fields and methods of it gets inherited by subclass. In this, the fields and methods are part of subclass, as if subclass declared itself.

Multiple Inheritance is a technique where one class extends more than one base class.

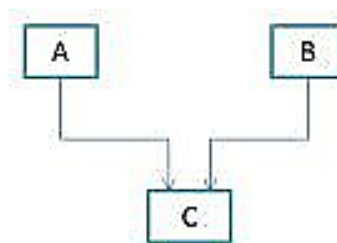


Fig 2.3 Multiple inheritance

The latter point is an advanced version of inheritance and is called multiple inheritance. Multiple inheritance provides an alternative approach where the new

class can inherit information from the two relevant classes. This is shown in the figure 2.4.

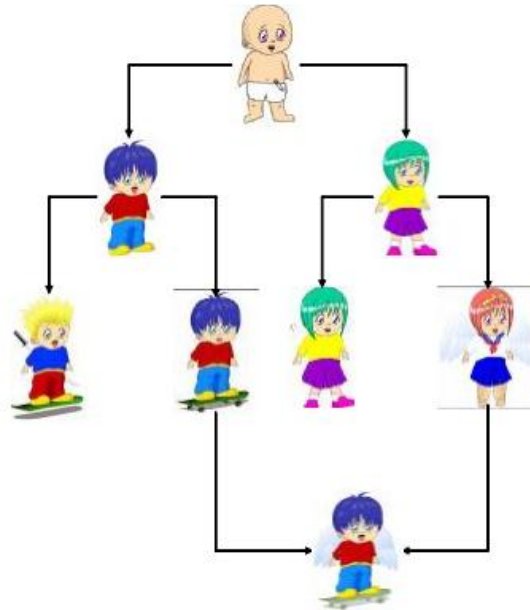


Fig 2.4 Multiple Inheritance

In the above figure we noticed some inconsistencies, where sloopy at the bottom has wheels and wings that are inherited from rim sloopy and float sloopy. In this, the boy is sloopy and not the girl and also wheels float sloopy inherited attributes of boy and girl which appear to be a conflict.

Check your progress 3

1. Select the correct option if class A is inherited by class B.
 - a. class B + class A
 - b. class B inherits class A
 - c. class B extends A
 - d. class B extends class A

2.5 Generalization as an Extension

Generalization is a process of defining a super class from a given set of semantically related entity set. Generalization is the process of extracting shared characteristics from two or more classes, and combining them into a generalized super class. Shared characteristics can be attributes, associations, or methods.

Generalization is the relationship between a class and one or more refined versions of it. The class being refined is called the “superclass” and each refined version is called a “subclass”. For example, Equipment is the superclass of Pump and tank. Attributes and operations that are common to a group of subclasses are attached to the superclass and shared by each subclass. Generalization is sometimes called an “is-a” relationship. Each instance of a subclass is an instance of the superclass.

Check your progress 4

1. What is generalization?
 - a. Relationship between one class and its refined versions
 - b. IS-A relationship
 - c. Defining super class from given set of related entities
 - d. All of these

2.6 Generalization as a Restriction

In generalization, an instance of a class is an instance of a class is an instance of all ancestors of the class. Therefore you can say that all ancestor class features must apply to the subclass instances. This includes not only the attributes on the ancestor classes but also the operations on the ancestor class.

A subclass may include many features, which is called as an extension. For example, fig.2.5 extends class Employee with three subclasses that inherit all Employee features and add new features of their own.

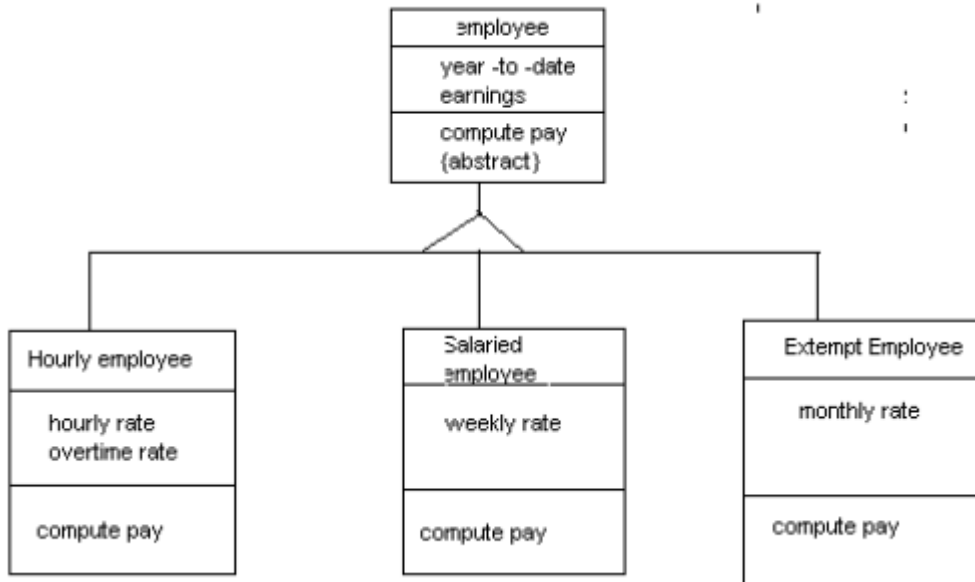


Fig 2.5 Class Employee

A subclass may also constrain ancestor attributes. This is called restriction because it restricts the values that instances can assume. For example, a circle is an ellipse that's major and minor axes are equal. Arbitrary changes to the attribute values of a restricted subclass may cause it to violate the constraints, such that the result no longer belongs to the original subclass. This is not a problem from the perspective of the super class because the result is still a valid super class instance.

Check your progress 5

1. What is meant by restriction in generalization?
 - a. Attribute
 - b. All ancestor class features must apply to the subclass instances, this is known as restriction
 - c. Inherited features
 - d. None of these

2.7 Metadata

Metadata is called as data about data or data that describes about other data. It is everywhere certainly where many data is useless without metadata as it is also data. Metadata summarizes basic information about data, which can make finding

and working with particular instances of data easier. In addition to document files, metadata is used for images, videos, spreadsheets and web pages. The use of metadata on web pages can be very important. Metadata for web pages contain descriptions of the page's contents, as well as keywords linked to the content. These are usually expressed in the form of metatags. The metadata containing the web page's description and summary is often displayed in search results by search engines, making its accuracy and details very important since it can determine whether a user decides to visit the site or not.

Check your progress 6

1. Which of the following is true about meta data?
 - a. It is the data that describes about other data
 - b. It summarizes basic information about data
 - c. It is expressed in the form of metatags
 - d. All of these

2.8 Constraints

A constraint could be a numeric or geometric relationship between objects. Constraints have a declarative nature. Constraints are a natural manner for describing relationships between objects. Combining constraint systems and object-oriented programming (OOP) seems hard. All existing systems implicitly compromise the encapsulation principle of OOP.

Constraints describe properties that have to be true at each moment in time for the entire system, without determining however they're to be preserved. The specification of constraints is mostly done by using informal text, operational restrictions or integrating constraints in existing model ideas.

Constraints are a way to specific general properties for the system, without specifying however they're being realized.

Check your progress 7

1. What can constitute constraints?
 - a. Numeric data
 - b. Geometric relationship
 - c. Both of these
 - d. None of these

2.9 An Object Model

An object model is typically a logical interface, software or system that's modelled through the utilization of object-oriented techniques. It permits the creation of an architectural software or system model before development or programming.

An object model is a component of the object-oriented programming (OOP) lifecycle.

Following are the benefits of using the object model are:

- It helps in quicker development of software.
- It is straightforward to maintain. Suppose a module develops an error, then a programmer will fix that specific module, whereas the other elements of the software are still up and running.
- It supports relatively hassle-free upgrades.
- It enables reuse of objects, designs, and functions.
- It reduces development risks, significantly in integration of advanced systems.

Check your progress 8

1. Why do there is a need to use object model?
 - a. It helps in faster development of software
 - b. reusability
 - c. reduces development risk
 - d. All of these

2.10 Let Us Sum Up

In this unit we have learnt that Inheritance can be defined as the process where one object acquires the properties of another. These inheritances make the information to manage in a hierarchical order. It is studied that a Subclass in Java is a method of inheritance which appears from Java superclass.

There exist three types of variables in Java as Local, Instance and Static. It is studied that local variable is a variable which is declared within method itself, instance variable is a variable which is declared within class but outside method and static variable is a variable which is declared as static is called static variable.

In Java, it is seen that constructor can be inherited as the constructor name is based on class name. While inheriting methods, the method signature should remain same.

2.11 Answers for Check Your Progress

Check your progress 1

Answers: (1 –d)

Check your progress 2

Answers: (1 –a)

Check your progress 3

Answers: (1 –c)

Check your progress 4

Answers: (1 –d)

Check your progress 5

Answers: (1 –b)

Check your progress 6

Answers: (1 –d)

Check your progress 7

Answers: (1 –c)

Check your progress 8

Answers: (1 –d)

2.12 Glossary

1. **Static variable** - It is a variable which is declared as static is called static variable
2. **Class** - These are group of objects with common properties and are like a blueprint from where objects are created.

2.13 Assignment

Write a short note on inheritance and its use.

2.14 Activities

Study abstract classes and its usage

2.15 Case Study

Discuss the output of this program?

```
class A {
    int i;
    int j;
    A() {
        i = 1;
        j = 2;
    }
}
class Output {
    public static void main(String args[])
    {
        A obj1 = new A();
        A obj2 = new A();
        System.out.print(obj1.equals(obj2));
    }
}
```

2.16 Further Readings

1. DOWLATSHAHI, S., 1992, Product design in a concurrent engineering environment: an optimization approach, *International Journal of Production Research*, 30(8), pp. 1803–1818.
2. ESCHENAUER, H., KOSKI, J. and OSY CZKA, A., 1990, *Multicriteria Design Optimization: Procedures and Applications* (New York: Springer-Verlag).
3. FENG, C. X. and KUSIAK, A., 1995, Constraint-based design of parts, *Computer-Aided Design*, 27(5), pp. 343–352.

Block Summary

In this block, you will be detailed with knowledge about Inheritance and its relationship with subclass and interface block. The concepts about types of variables with illustrations are explained in step by step manner. The block stress on information related to packages and interface with examples about inherit abstract methods of interface. The information related to declaring a reference variable of interface is well explained with examples.

After studying this block, you will feel self confident while working on simple Java platform and can enhance their knowledge by studying certain examples and illustrations mentioned in this block. With such detailed knowledge on Inheritance, Interface, Packages and Exceptions in Java, you can be benefitted in future.

Block Assignment

Short Answer Questions

1. What is object oriented modeling?
2. Explain inheritance and its types
3. What is the difference between generalisation and specialisation?
4. Write short note on Abstract classes

Long Answer Questions

1. What is multiple inheritance. Explain in detail with the help of an example
2. Write short note on metadata.
3. Write note on generalisation as a restriction

Enrolment No.

1. How many hours did you need for studying the units?

Unit No	1	2	3	4
Nos of Hrs				

2. Please give your reactions to the following items based on your reading of the block:

Items	Excellent	Very Good	Good	Poor	Give specific example if any
Presentation Quality	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	_____ _____
Language and Style	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	_____ _____
Illustration used (Diagram, tables etc)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	_____ _____
Conceptual Clarity	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	_____ _____
Check your progress Quest	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	_____ _____
Feed back to CYP Question	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	_____ _____

3. Any Other Comments

.....

.....

.....

.....

.....

.....

.....

.....



“

*Education is something
which ought to be
brought within
the reach of every one.*

”

- Dr. B. R. Ambedkar



Dr. Babasaheb Ambedkar Open University
'Jyotirmay Parisar', Opp. Shri Balaji Temple, Sarkhej-Gandhinagar Highway, Chharodi,
Ahmedabad-382 481.

OBJECT ORIENTED ANALYSIS AND DESIGN

PGDCA -204

BLOCK 2:
OBJECT ORIENTED ANALYSIS
AND SYSTEM DESIGN

Dr. Babasaheb Ambedkar Open University
Ahmedabad



OBJECT ORIENTED ANALYSIS AND DESIGN



Knowledge Management and
Research Organization
Pune



Editorial Panel

Author

PROF. K J Sharma

Language Editor

Mr. Vipul Shelke

Graphic and Creative Panel

Ms. K. Jamdal

Ms. Lata Dawange

Mr. Prashant Tikone

Copyright © 2015 Knowledge Management and Research Organization.

All rights reserved. No part of this book may be reproduced, transmitted or utilized in any form or by means of, electronic or mechanical, including photocopying, recording or by any information storage or retrieval system without written permission from us.

Acknowledgment

Every attempt has been made to trace the copyright holders of material reproduced in this book. Should an infringement have occurred, we apologize for the same and will be pleased to make necessary correction/amendment in future edition of this book.

The content is developed by taking reference of online and print publications that are mentioned in Bibliography. The content developed represents the breadth of research excellence in this multidisciplinary academic field. Some of the information, illustrations and examples are taken "as is" and as available in the references mentioned in Bibliography for academic purpose and better understanding by learner.'



ROLE OF SELF INSTRUCTIONAL MATERIAL IN DISTANCE LEARNING

The need to plan effective instruction is imperative for a successful distance teaching repertoire. This is due to the fact that the instructional designer, the tutor, the author (s) and the student are often separated by distance and may never meet in person. This is an increasingly common scenario in distance education instruction. As much as possible, teaching by distance should stimulate the student's intellectual involvement and contain all the necessary learning instructional activities that are capable of guiding the student through the course objectives. Therefore, the course / self-instructional material are completely equipped with everything that the syllabus prescribes.

To ensure effective instruction, a number of instructional design ideas are used and these help students to acquire knowledge, intellectual skills, motor skills and necessary attitudinal changes. In this respect, students' assessment and course evaluation are incorporated in the text.

The nature of instructional activities used in distance education self-instructional materials depends on the domain of learning that they reinforce in the text, that is, the cognitive, psychomotor and affective. These are further interpreted in the acquisition of knowledge, intellectual skills and motor skills. Students may be encouraged to gain, apply and communicate (orally or in writing) the knowledge acquired. Intellectual-skills objectives may be met by designing instructions that make use of students' prior knowledge and experiences in the discourse as the foundation on which newly acquired knowledge is built.

The provision of exercises in the form of assignments, projects and tutorial feedback is necessary. Instructional activities that teach motor skills need to be graphically demonstrated and the correct practices provided during tutorials. Instructional activities for inculcating change in attitude and behavior should create interest and demonstrate need and benefits gained by adopting the required change. Information on the adoption and procedures for practice of new attitudes may then be introduced.

Teaching and learning at a distance eliminates interactive communication cues, such as pauses, intonation and gestures, associated with the face-to-face method of teaching. This is particularly so with the exclusive use of print media. Instructional activities built into the instructional repertoire provide this missing interaction between the student and the teacher. Therefore, the use of instructional activities to affect better distance teaching is not optional, but mandatory.

Our team of successful writers and authors has tried to reduce this.

Divide and to bring this Self Instructional Material as the best teaching and communication tool. Instructional activities are varied in order to assess the different facets of the domains of learning.

Distance education teaching repertoire involves extensive use of self-instructional materials, be they print or otherwise. These materials are designed to achieve certain pre-determined learning outcomes, namely goals and objectives that are contained in an instructional plan. Since the teaching process is affected over a distance, there is need to ensure that students actively participate in their learning by performing specific tasks that help them to understand the relevant concepts. Therefore, a set of exercises is built into the teaching repertoire in order to link what students and tutors do in the framework of the course outline. These could be in the form of students' assignments, a research project or a science practical exercise. Examples of instructional activities in distance education are too numerous to list. Instructional activities, when used in this context, help to motivate students, guide and measure students' performance (continuous assessment)



PREFACE

We have put in lots of hard work to make this book as user-friendly as possible, but we have not sacrificed quality. Experts were involved in preparing the materials. However, concepts are explained in easy language for you. We have included many tables and examples for easy understanding.

We sincerely hope this book will help you in every way you expect.

All the best for your studies from our team!



OBJECT ORIENTED ANALYSIS AND DESIGN

Contents

BLOCK 1: OBJECT ORIENTED MODELLING

UNIT 1 INTRODUCTION TO OBJECT ORIENTED MODELLING

Object Oriented Modeling, Characteristics Object Oriented Modeling: Class and Objects, Links and Association, Generalization and Inheritance, An Object Model. Benefits of OO Modeling, Introduction to OOAD tools,

UNIT 2 ADVANCE MODELING CONCEPTS

Aggregation, Abstract Class, Multiple Inheritance, Generalization as an Extension, Generalization as a Restriction, Metadata, Constraints, An Object Model

BLOCK 2: OBJECT ORIENTED ANALYSIS AND SYSTEM DESIGN

UNIT 1 OBJECT ORIENTED ANALYSIS

Object Oriented Analysis, Problem Statement: an Example, Differences between Structured Analysis and Object Oriented Analysis, Analysis Techniques: Object Modeling, Dynamic Modeling, and Functional Modeling. Adding Operations. Analysis Iteration

UNIT 2 SYSTEM DESIGN

System Design: An Object Oriented Approach, Breaking into Subsystems, Concurrency Identification, Management of data store, Controlling events between Objects, Handling Boundary Conditions



BLOCK 3: OBJECT DESIGN AND DYNAMIC MODELING

UNIT 1 OBJECT DESIGN

Object Design for Processing, Object Design Steps, Designing a Solution, Choosing Algorithms, Choosing Data Structures, Defining Classes and delegation of Responsibilities to Methods

UNIT 2 DYNAMIC MODELING

Events, State and State Diagram, Elements of State Diagrams, Examples of State Diagrams, Advance Concepts in Dynamic Modeling, Concurrency, A Dynamic model

BLOCK 4: FUNCTIONAL MODELING AND UML

UNIT 1 FUNCTIONAL MODELING

Functional Models, Data Flow Diagrams, Features of a DFD, Design flaws in DFD, A Functional model, Relationship between Object, Dynamic, and Functional Models

UNIT 2 USING UML

UML: Introduction, Object Model Notations: Basic Concepts, Structural Diagrams: Class, Object, Composite, Package, Component, Deployment. Behavioral Diagrams: Use Case, Communication, Sequence, Interaction Overview, Activity, State. Modeling with Objects

UNIT 3 CASE STUDY

This unit will cover all the OOAD aspects Covered in previous units of this course.



Dr. Babasaheb
Ambedkar
Open University

PGDCA-204

OBJECT ORIENTED ANALYSIS AND DESIGN

BLOCK 2: OBJECT ORIENTED ANALYSIS AND SYSTEM DESIGN

UNIT 1

OBJECT ORIENTED ANALYSIS

03

UNIT 2

SYSTEM DESIGN

14

BLOCK 2: OBJECT ORIENTED ANALYSIS AND SYSTEM DESIGN

Block Introduction

Object oriented design is totally contrary to both structured methodologies and information engineering. "Both structured design and information engineering use function-oriented decomposition rules, resulting in a structure of procedure-oriented program modules that contain programs, subroutines, and functions with solely procedural code.

In this block, we will detail about the basic of Analysis Techniques such as Object Modelling, Dynamic Modelling and Functional Modelling. The block will focus on the study and concept of Structured Analysis and Object Oriented Analysis. You will show comparison about Structured Analysis and Object Oriented Analysis.

In this block, you will make to learn and understand about the Object oriented analysis design approach supporting system as group of objects in terms of logical frame system. The concept related to Object-oriented analysis with groups items interaction of class, data or behaviour also explained to you.

Block Objective

After learning this block, you will be able to understand:

- Understanding of object oriented design;
- Knowledge about breaking of system to subsystems;
- Features of software design
- Identifying handle concurrency;
- Features of object oriented analysis
- Idea about Structured Analysis and Object Oriented Analysis
- Understanding of Analysis Techniques

Object Oriented
Analysis and
System Design

Block Structure

Unit 1: Object Oriented Analysis

Unit 2: System Design

UNIT 1: OBJECT ORIENTED ANALYSIS

Unit Structure

1.0 Learning Objectives

1.1 Introduction

1.2 Object Oriented Analysis

1.3 Problem Statement: an Example

1.4 Differences between Structured Analysis and Object Oriented Analysis

1.5 Analysis Techniques: Object Modeling, Dynamic Modeling, and Functional Modeling

1.6 Adding Operations

1.7 Analysis Iteration

1.8 Let Us Sum Up

1.9 Answer for Check Your Progress

1.10 Glossary

1.11 Assignment

1.12 Activities

1.13 Case Study

1.14 Further Readings

1.0 Learning Objectives

After learning this unit, you will be able to understand:

- About Object Oriented Analysis
- About Structured Analysis
- About Object Modeling
- About Functional Modeling

1.1 Introduction

In the system analysis or object-oriented analysis part of software development, the system needs are determined, the classes are identified and the relationships among classes are known. The 3 analysis techniques that are used in conjunction with one another for object-oriented analysis are object modelling, dynamic modelling, and functional modelling.

1.2 Object Oriented Analysis

Object oriented analysis related to the methodology which combines the item that mix with other in terms of class, data or behaviour, thereby creating a model which perfectly shows the required purpose of system. Object-oriented analysis will not factor on the implementation, limitations in the model.

The advantage of using object oriented methodology is to create what you already contain. With this approach, the ability to use the code any time and develop additional maintainable systems in lesser amount of time is possible. Additionally, object-oriented systems are better designed, more resilient to change, and more reliable, since they're built from fully tested and debugged classes.

Check your progress 1

1. Object Oriented analysis includes_____.

 - a. Object modeling
 - b. Dynamic modeling
 - c. Functional modeling.
 - d. All of these

1.3 Problem Statement: an Example

You must understand that you are looking for a statement of requirements, not a proposal for a solution. OOA specifies the structure and the behaviour of the object that comprises the requirements of that specific object. Different types of models are required to specify the requirements of the objects. These object

models contain the definition of objects in the system, which includes: the name of object, its attributes, and its relation with other object. As you know, an object is a representation of a real-life entity, or an abstraction. For example, objects in a flight reservation system might include: an airplane, an airline flight, an icon on a screen, or even a full screen with which a travel agent interacts. The behaviour, or state model describes the behaviour of the objects in terms of the states of the object and the transitions allowed between objects, and the events that cause objects to change states.

These models are created and maintained using CASE tools that support the illustration of objects and object behaviour.

You can say that the problem statement could be a general description of the user's difficulties, desires, and its purpose is to identify the overall domain within which you will be operating, and give some plan of why you're doing OOA.

It is vital for an analyst to separate the true necessities from design and implementation decisions.

Problem statement shouldn't be incomplete, ambiguous, and inconsistent. Attempt to state the requirements precisely, and to the purpose. Do not make it cumbersome. Some requirements appear reasonable but don't work. These kinds of requirements should be identified. the aim of the subsequent analysis is to fully understand the problem and its implications, and to bring out the true intent of the client.

Check your progress 2

1. What can be the objects in a library management system?
 - a. Books
 - b. Library
 - c. Student
 - d. All of these

1.4 Differences between Structured Analysis and Object Oriented Analysis

Object oriented analysis design (OOAD) is essentially a bottom up approach that supports viewing the system as a collection of components (objects) which will be logically kept together to create the system.

Advantages and disadvantages of Object oriented Analysis and design

Advantages:

The OO approach inherently makes each object a standalone component which will be reused not only within a particular stat problem domain; however are also completely different downside domains, having the requirement of similar objects.

The other main advantage of object oriented (OO) is that the focus on data relationships. You cannot develop a successful system where data relationships aren't well understood. An OO model provides all of the insight of an ER diagram and contains additional information related to the ways to be performed on the data.

Disadvantages:

You know that OO methods only build functional models within the objects. There's no place within the methodology to build a complete functional model. Whereas this is not a problem for a few applications (e.g., building a software toolset), except for large systems, it will lead to missed needs.

Another disadvantage of the object oriented analysis design (OOAD) is in system modelling for performance and sizing. The object oriented (OO) models don't simply describe the communications between objects. Indeed, a basic concept of object oriented (OO) is that the object needn't know who is invoking it. Whereas this results in a flexible design, performance modelling can't be handled simply.

The object oriented (OO) analysis design itself doesn't give support for identifying that objects can generate an optimal system design. Specifically, there's no single diagram that shows all of the interfaces between objects.

Advantages and disadvantages of Structured Analysis

With expertise, you'll come to know to grasp that almost all customers understand structured methods better than object oriented (OO) ways. Since one of the main reasons of modelling a system is for communication with customers

and users, there's an advantage in providing structured models for information exchange with user groups or customers.

The disadvantage with structured methods is that they do not readily support the use of reusable modules. The top down method works well for new development, however doesn't give the mechanisms for “designing in” the use of existing components. The top down process of functional decomposition doesn't lead to a set of requirements that map well to existing components.

Check your progress 3

1. Object oriented approach is useful because?
 - a. It supports reusability of modules
 - b. It is top – down approach
 - c. It can handles performance modeling
 - d. It describes the communication between objects

1.5 Analysis Techniques: Object Modelling, Dynamic Modelling, and Functional Modelling

Object Modelling Technique is a methodology that deals with object-oriented development in analysis and design phases. In analysis phase, a problem statement carries list of goals and key ideas within a domain. Problem statement seems in three views or models:

- Object model
- Dynamic model
- Functional model

An object model shows artifacts of system showing static and stable phenomena in model domain. The idea behind it's classes and associations with attributes and operations. Aggregation and generalization are predefined relationships

The notations used in object model are shown in fig 1.1

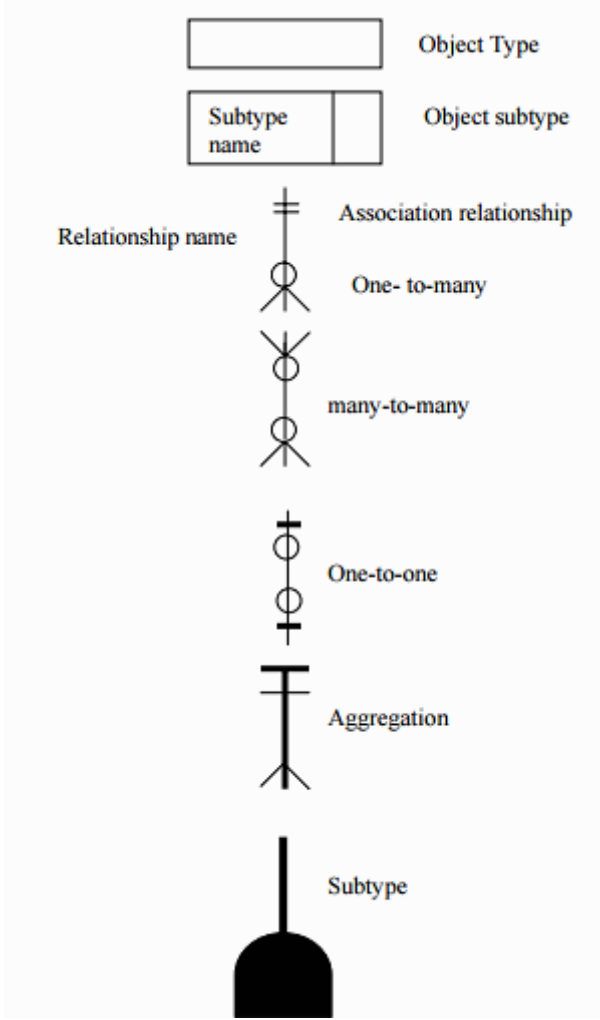


Fig 1.1 Notations in Object Model

The dynamic model shows interaction among such artifacts that shows events, states and transitions. It shows a state/transition view on model having states, transitions that exist among states and events.

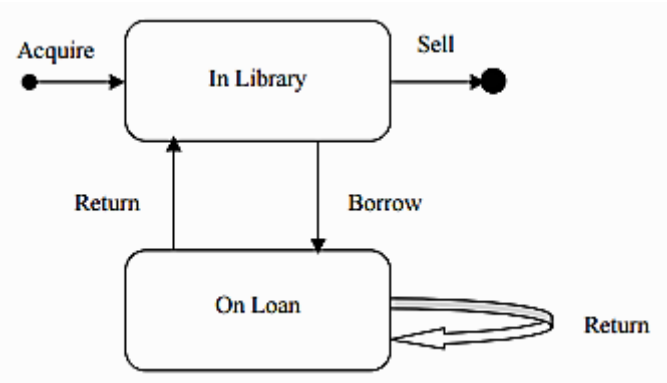


Fig 1.2 State Diagrams

In this, the actions are modelled inside the states. The notations used in this modelling is shown in fig 1.3

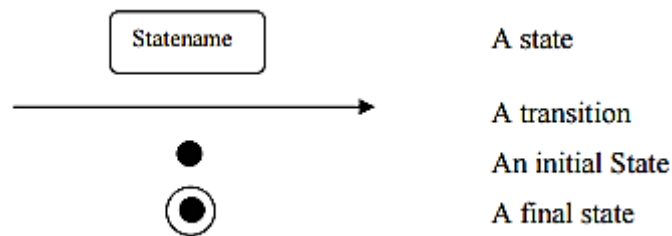


Fig 1.3 Notations

The functional model carries system ways from perspective of data flow. It handles the process viewpoint of model that corresponds to data flow diagrams. It carries main ideas related to process, data store, data flow and actors. Steps employed in framing functional Model comprises of:




- Finding input/output values
- Creating data flow diagrams with functional dependencies
- Explaining the functions
- Locating constraints
- Showing rules of optimisation

The system design phase moves along with analysis phase where architecture is established and is arranged in subsystems that allocates processes and tasks with account concurrency and collaboration.

The object design phase is with system design phase wherever plan gets implemented. Object classes are carried out with algorithms having attention to optimize path to persistent data.

Check your progress 4

1. Which notation is used to describe the initial state of an object in object modeling?

- a. 
- b. 
- c. 
- d. None of these

1.6 Adding Operations

Whenever you examine the operations in OOPs you find queries regarding attributes or associations within the object model, to events within the dynamic model and functions within the functional model. You can see operations from the object model. It includes reading and writing attribute values and association links. Within the object model operations are presented with an attribute. During analysis nothing is private, you assume that all attributes are accessible.

During analysis events are sent to the target objects. An operation on the object is presented as labels on transitions and should not be explicitly listed in the object model. The functions should be easy and summarized on the object model. Organise the functions into operations on objects.

Check your progress 5

1. What is the use of == operator?
 - a. Checks if the values of two operands are equal or not
 - b. Checks if the values of two operands are equal or not
 - c. Checks if the value of left operand is greater than or equal to the value of right operand,
 - d. None of these

1.7 Analysis Iteration

To understand any problem properly you have to repeat the task which means that analysis needs repetition. First, simply get the overview of the problem, build a rough draft, and then, reiterate the analysis according to your understanding. Finally, you must verify the final analysis with the client or application domain experts. During the iteration processes refining analysis and restating of requirement takes place.

Refining the ratio Analysis

Basically, refinement leads to purity. Thus to get a cleaner, more understandable and coherent design, you would like to reiterate the analysis process.

- Re-examine the model to remove inconsistencies, and imbalances with and across the model.
- Remove anomaly, and wrong ideas from the model.
- Refining sometimes involves restructuring the model.
- Define constraints within the system in a very better way.
- You should keep track of generalizations factored on the wrong attributes.
- Include exceptions within the model, many special cases, lack of expected symmetry, and an object with two or more sets of unrelated attributes, or operations.
- You should remove extra objects or associations from the model. Also, take care to remove redundancy of objects and their extra attributes.

Restating the necessities

To have clarity of the analytical model of the system you must state the necessities specific performance constraints with the optimization criteria in one document verify within the different document. You'll state the method of a solution. Verify the ultimate model with the client. The requirement analysis should be confirmed and clearly understood by the client.

Check your progress 6

1. What is Iteration?
 - a. Repetition of design process
 - b. Reusability
 - c. Re-examine the model to remove inconsistencies
 - d. All of these

1.8 Let Us Sum Up

Object oriented analysis (OOA) describes what the customer requires, it establishes as basis for the creation of a software design, and it defines a collection of needs which will be validated. You also remember that the requirement analysis should be designed in such the simplest way that it should tell you what to be done, not how it's implemented.

The object model is that the principal output of an analysis and design process. Dynamic modelling is elaborated further by adding the concept of time: new attributes are computed as a function of attribute changes over time.

1.9 Answers for Check Your Progress

Check your progress 1

Answers: (1 –d)

Check your progress 2

Answers: (1 –d)

Check your progress 3

Answers: (1 –a)

Check your progress 4

Answers: (1 –a)

Check your progress 5

Answers: (1 –a)

Check your progress 6

Answers: (1 –c)

1.10 Glossary

1. **Object-oriented analysis** - Process having packed with teems which mixup with by class, data or behaviour for modeling purpose.

1.11 Assignment

Explain how you can define an object model of a system.

1.12 Activities

Give an example of operations from State Actions and Activities.

1.13 Case Study

Show the basic dynamic model of telephone.

1.14 Further Readings

1. Norman, Ronald- object oriented system analysis and design –prentice hall 1996.
2. Coad. P and Yourdon .E – “Object Oriented Analysis” – Yourdon press.
3. Coad. P and Yourdon .E – “Object Oriented Design” – Yourdon press.
4. Rambaugh, James, Michael –Object oriented Modelling and design.

UNIT 2: SYSTEM DESIGN

Unit Structure

- 2.0 Learning Objectives**
- 2.1 Introduction**
- 2.2 System Design: An Object Oriented Approach**
- 2.3 Breaking into Subsystems**
- 2.4 Concurrency Identification**
- 2.5 Management of data store**
- 2.6 Controlling events between Objects**
- 2.7 Handling Boundary Conditions**
- 2.8 Let Us Sum Up**
- 2.9 Answers for Check Your Progress**
- 2.10 Glossary**
- 2.11 Assignment**
- 2.12 Activities**
- 2.13 Case Study**
- 2.14 Further Readings**

2.0 Learning Objectives

After learning this unit, you will be able to understand:

- Overview of object oriented design
- Breaking down system to subsystems
- Explain about software design as set of interacting objects managing states and operation
- Identify concurrent object, and explain how to handle concurrency

2.1 Introduction

System design is that the process of defining the components, modules, interfaces, and data for a system to satisfy specified requirements. System development is that the process of creating or altering systems, along with the processes, practices, models, and methodologies used to develop them. Object oriented design (OOD) is concerned with developing an object oriented model of a software system to implement the identified needs. Several OOD methods are delineated since the late 1980s.

2.2 System Design: An Object Oriented Approach

If design stage is a set of objects with their responsibilities and collaborators. These objects will most likely become classes, wherever the actual objects in the system are going to be instances of these classes. By creating the item definitions into classes, we will create as many of the object as we want and handle growth and complexity within the system. The outcome of the design stage could be a description detailed enough to come from. The design carries 2 parts:

- A {class|a category} diagram defining the attributes and services of {each of every} class and formally identifying the connections between each class.
- A detailed description of what each service is supposed to try and do.

Software systems designed with structured design methodology don't support some of the required quality attributes like reusability, portability and mapping to the problem domain. Many large organisations find that systems designed with structured approaches are less reusable and fewer maintainable than those designed with object-oriented approaches.

- Although object oriented methods have roots that are powerfully anchored back within the 60s, structured and functional methods were the first to be used. This is not very uncommon, since functional methods are inspired directly by computer architecture (a proven domain, standard to computer scientists). The separation of data and program as exists physically within the hardware was translated into functional methods. This is the reason why computer scientists got into the habit of thinking in terms of system functions. Later it absolutely was felt that hardware should act as the servant of the software that's executed on it rather than imposing architectural constraints on system development method.

Object Oriented Analysis and System Design

- Traditional System Analysis and design: traditional System Analysis and design (SAD) has 3 basic life cycle models. A typical computer code lifecycle consists of following phases:
- Planning
- Development
- Maintenance

Key Criteria

- a well-defined methodology
- traceability among steps
- documentation
- control

Software Life Cycle

The software design consists of a number of stages, or phases

- Requirement Phase– determine use need
- Specification Phase– define requirement
- Design Phase: design the system
- Implementation Phase– fabricate the system
- Testing and Integration Phase– testing the fabricated system
- Maintenance phase
- Retirement

An object-oriented design process

1. Define the context and modes of Object Oriented Design
2. Designs the system architecture
3. Identifies the principal system objects
4. Identifies concurrency in the problem
5. Handling boundary conditions
6. Develops design models
7. Specifies object interfaces

Check your progress 1

1. Which phase of software development life cycle involves designing of system?
 - a. Design phase
 - b. Implementation phase
 - c. Requirement phase
 - d. None of these

2.3 Breaking into Subsystems

Object-oriented decomposition aims at identifying individual autonomous objects that encapsulate both a state and a certain behaviour. Each major component of the system is named a subsystem. Then communication among these objects leads to the specified solutions. Decomposition of system in function-data model (SAD) and object oriented decomposition is given in fig 2.1.

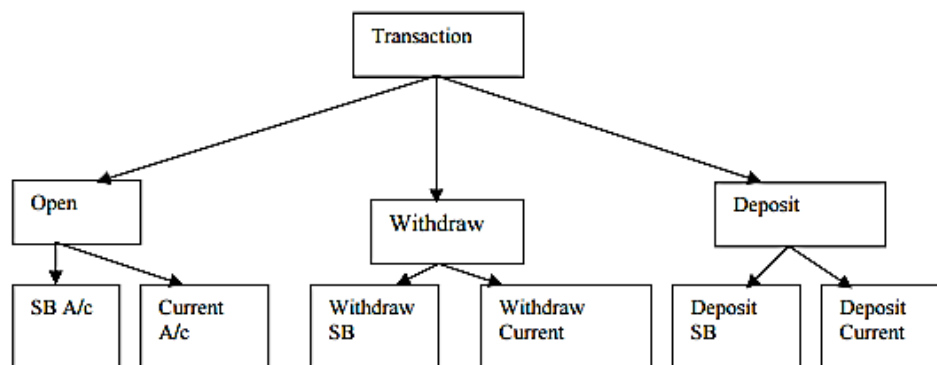


Fig 2.1 Functional Data Decomposition

Although each solution facilitate deal with complexity, we've got reasons to believe that an object-oriented decomposition is favourable because, the object-oriented approach provides for a semantically richer framework that leads to decompositions that are more closely related to entities from the real world. Moreover, the identification of abstractions supports heaving (more abstract) solutions to be reused, and therefore the object-oriented approach supports the evolution of systems better, as those ideas that are additional seemingly to alter can be hidden within the objects.

Object-oriented decompositions of systems tend to be better able to deal with modification. Each subsystem incorporates a well-defined interface that

communicates with rest of the system. Each of these interfaces defines all form of interaction that is needed for proper functioning of the whole system; however the internal implementations are left to the sub-system itself. This can be because they manage to encapsulate those items that are likely to alter (such as functionality, sequence of behaviour and attributes) within an object and hide them from the outside world. The advantage is that the outside cannot see them, and therefore cannot be dependent on them and does not need to be changed if these items change. Also, object-oriented decompositions are closer to the problem domain, as they directly represent the real-world entities in their structure and behaviour. The abstraction primitives built into reuse have a huge potential of reuse as commonalities between similar objects is factored out, and then, the solutions is reused. Finally, object-orientation has the advantage of continuity throughout analysis, design implementation, and persistent representation.

Advantages of Decomposition

- Separate people can work on each subsystem.
- Every software engineer must specialize in a domain.
- Every individual component is smaller, and therefore easier to understand and manage.
- Part of the subsystem can be replaced or changed without having replaced or extensively changed the other subsystems.

Check your progress 2

1. Why decomposition of system is required?
 - a. Sub systems are able to deal with modifications efficiently
 - b. Each subsystem incorporates a well-defined interface that communicates with rest of the system
 - c. Small components are easy to manage
 - d. All of these

2.4 Concurrency Identification

Concurrency in objects may be identified by the manner they change state. Current objects will change state independently. Aggregation implies concurrency. Concurrency in OOAD study is described and studied in dynamic modelling. If the objects are concurrent in nature we've to assign them to, different thread of control. For instance, withdraw and deposit operations related to a bank account is also executed in parallel (concurrently).

Concurrency is identified in a dynamic model. 2 objects are said to be concurrent (parallel) if they will receive events at the same time. Concurrent objects are required to be assigned to different threads of control.

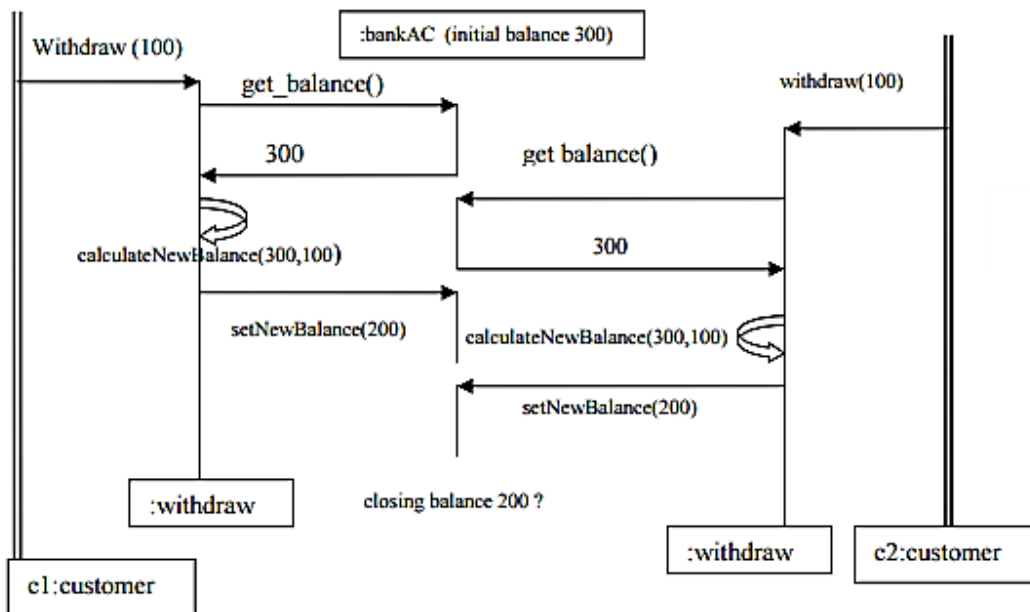


Fig 2.3 Concurrency without synchronisation

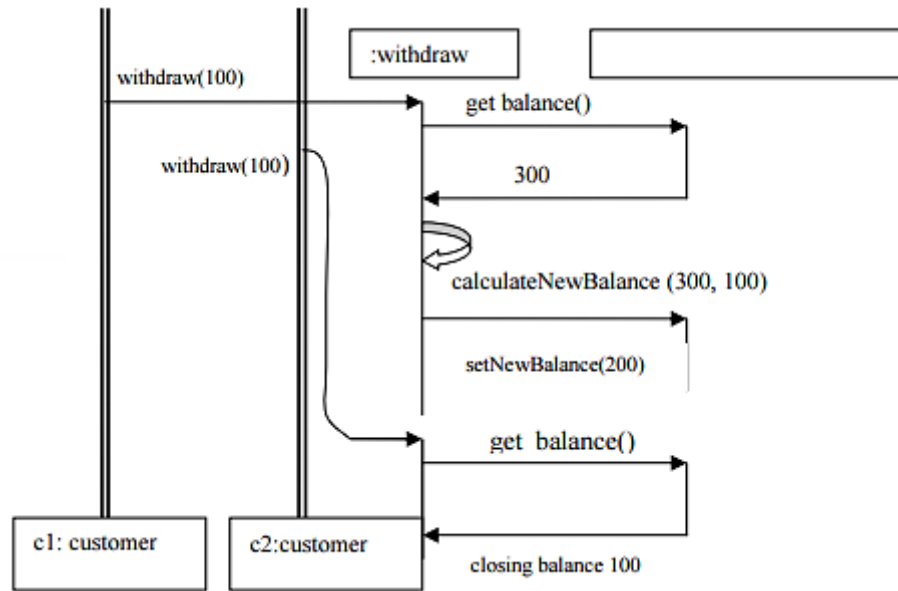


Fig 2.4 Concurrency with synchronisation

Check your progress 3

1. Two objects are said to be concurrent_____.
 - a. When they will receive events concurrently
 - b. If two objects are same
 - c. If they receive same events
 - d. None of these

2.5 Management of data store

Some objects within the models need to be persistent, to store the state of the object permanently in database. Most systems need persistent data that isn't lost when the system closes down. These data are stored in file system, in a or database. Object-oriented applications often use relational databases to provide persistence.

Designer needs to:

1. Determine what data needs to be persistent
2. Design a suitable database schema for the database.

In most of the cases, persistent class maps to one relational table (leaving aside the inheritance issue, for the moment). Within the simplest model, a class

from the logical model maps to a relational table, either in whole, or in part. The logical extension of this can be that a single object (or instance of a class) maps to a single table row. Persistent object is stored with one of the following:

Files

- Cheap, simple, permanent storage
- Low level (Read, Write)
- Applications should add code to provide a suitable level of abstraction.

Relational database

- Standardized, easy to port
- Supports multiple writers and readers
- Mature technology

Object database

- One-to-one mapping from the analysis model
- Associations are directly represented
- Slower than relational databases for complex queries

Issues once selecting a database

- Storage space requirement: A database requires about triple the storage space of actual information.
- Response time: The response time of the database for I/O or communication bound (in case of distributed databases) request.
- Locking modes pessimistic locking: Lock before accessing an object and unharnessed once object access is complete.

Optimistic locking: read and writes might freely occur (high degree of concurrency).

- Administration: trendy software system needs specially trained support staff to line up security policies, manage the disk space, prepare backups, fine-tune the performance, and monitor performance.

Check your progress 4

1. What is optimistic locking mode?
 - a. It is a Lock that is used before accessing an object
 - b. It is a high degree of concurrency lock
 - c. It restricts read and write operations
 - d. None of these

2.6 Controlling events between Objects

Examples of events are click and flight leaving from an airport. An event doesn't have a fixed duration. Each factor that happens modelled as an event. After an event, objects change their state, and these are represented by a state diagram.

Events are classified as four types in UML

1. Signals
2. Calls
3. Passing of time
4. Change in State

Events additionally include inputs, decisions, interrupts and actions performed by users or any external device. Every event has a sender and receiver. In most of the cases the sender and receives are the same object. A state without a predecessor and successor are ambiguous, and care should be taken to represent initiations and termination of events. Events that have same effect on the control flow must be grouped together though their value differs. The events are to be allocated to the object classes that send/receive it.

Most of the look issues of systems are concerned with steady-state behaviour. However, the system design phase should additionally address the initiation and finalization of the system. This is often addressed by a set of new uses cases referred to as administration use cases. Now, let us discuss however these problems are handled.

Check your progress 5

1. Unified Modeling Language(UML) classifies events into_____.
- Signals
 - Calls
 - passing of states
 - All of these

2.7 Handling Boundary Conditions

These are some conditions that to be handled in any system initialization and termination

- Describes however the system is brought from a non-initialized state to steady-state. It describes normal operations like start-up, shutdown, reconfiguration, restart/reset, backup, etc.
- Describes what resources are clean up, and that systems are notified upon termination.
- Describes however the system is adapted to a local installation. Failure
- Unplanned termination of the system.

Many possible causes: failure of system hardware, bugs, operator errors, external problems.

- Sensible system design foresees fatal failures.

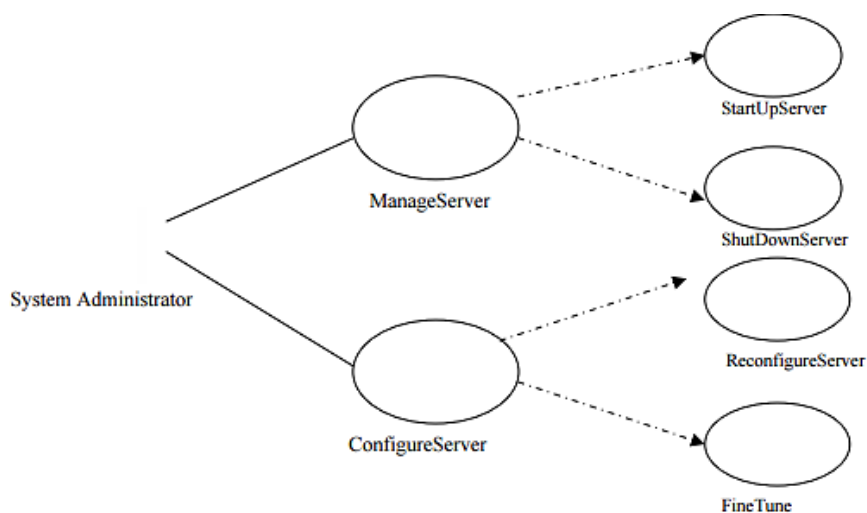


Fig 2.5 Boundary conditions

Check your progress 6

1. What is the need of using boundary conditions?
 - a. It describes normal operations like start-up, shutdown etc.
 - b. It describes how the system is adapted to a local installation
 - c. It describes termination use cases
 - d. All of these

2.8 Let Us Sum Up

In this unit we have learnt that OOD is a technique which is helpful for development of huge and complex systems. It is seen that moving from functional approach to object-oriented needs a translation of functional model elements into object model elements that is easy.

It is noted that high-level system style approach involves breaking down of system into easy and comparatively independent sub systems for higher manageability. Concurrency is inherent to objects, and concurrent objects can't be folded into one thread of control. Concurrency should be dealt with during the design method as dealing with concurrency after the system is implemented is difficult.

Data are often stored in flat files or database management system. Object-oriented databases provide support for all fundamental object modelling concepts like classes, Attributes, Methods. Associations and Inheritance are object-oriented databases which lack theoretical foundation and standards which is concerned with system design in handling boundary conditions.

2.9 Answers for Check Your Progress

Check your progress 1

Answers: (1 –a)

Check your progress 2

Answers: (1 –d)

Check your progress 3

Answers: (1 –a)

Check your progress 4

Answers: (1 –b)

Check your progress 5

Answers: (1 –d)

Check your progress 6

Answers: (1 –d)

2.10 Glossary

1. **Object oriented analysis design** - A bottom up approach which supports system as group of objects that is logical to frame the system.

2.11 Assignment

What do you mean by Concurrency in objects?

2.12 Activities

What are the advantages of decomposing a system?

2.13 Case Study

Discuss the broad steps of an object-oriented design process.

2.14 Further Readings

1. O.M. Nierstrasz, A Survey of Object-Oriented Concepts.
2. In W. Kim, F. Lochovsky (eds.): Object-Oriented Concepts

Block Summary

In this block, you have learnt and understand about the basic of software design and identification of concurrency. The block gives an idea on the study and concept of object oriented analysis design with detailed about support approach in terms of grouping of objects. You have been well explained on the concepts of structured analysis and object oriented analysis.

The block detailed about the basic of various analysing techniques such as object modelling, dynamic modelling and functional modelling. The concept related to object oriented analysis design bottom up approach in terms of system support are well explained to students. You will be demonstrated practically about Object Modelling.

Block Assignment

Short Answer Questions

1. Explain Iteration Analysis?
2. Define the steps used in constructing Functional Model?
3. What are the shortcomings in structured approach?
4. Differentiate between object oriented decomposition and structured decomposition?

Long Answer Questions

1. How do you see the final model after iterative analysis?
2. What are the benefits of OOA technology?
3. Explain how you can define an object model of a system.

Enrolment No.

1. How many hours did you need for studying the units?

Unit No	1	2	3	4
Nos of Hrs				

2. Please give your reactions to the following items based on your reading of the block:

Items	Excellent	Very Good	Good	Poor	Give specific example if any
Presentation Quality	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	_____ _____
Language and Style	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	_____ _____
Illustration used (Diagram, tables etc)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	_____ _____
Conceptual Clarity	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	_____ _____
Check your progress Quest	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	_____ _____
Feed back to CYP Question	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	_____ _____

3. Any Other Comments

.....

.....

.....

.....

.....

.....

.....

.....



“

*Education is something
which ought to be
brought within
the reach of every one.*

”

- Dr. B. R. Ambedkar



Dr. Babasaheb Ambedkar Open University
'Jyotirmay Parisar', Opp. Shri Balaji Temple, Sarkhej-Gandhinagar Highway, Chharodi,
Ahmedabad-382 481.

OBJECT ORIENTED ANALYSIS AND DESIGN

PGDCA -204

**BLOCK 3:
OBJECT DESIGN AND DYNAMIC
MODELING**

**Dr. Babasaheb Ambedkar Open University
Ahmedabad**



OBJECT ORIENTED ANALYSIS AND DESIGN



Knowledge Management and
Research Organization
Pune



Editorial Panel

Author

PROF. K J Sharma

Language Editor

Mr. Vipul Shelke

Graphic and Creative Panel

Ms. K. Jamdal

Ms. Lata Dawange

Mr. Prashant Tikone

Copyright © 2015 Knowledge Management and Research Organization.

All rights reserved. No part of this book may be reproduced, transmitted or utilized in any form or by means of, electronic or mechanical, including photocopying, recording or by any information storage or retrieval system without written permission from us.

Acknowledgment

Every attempt has been made to trace the copyright holders of material reproduced in this book. Should an infringement have occurred, we apologize for the same and will be pleased to make necessary correction/amendment in future edition of this book.

The content is developed by taking reference of online and print publications that are mentioned in Bibliography. The content developed represents the breadth of research excellence in this multidisciplinary academic field. Some of the information, illustrations and examples are taken "as is" and as available in the references mentioned in Bibliography for academic purpose and better understanding by learner.'



ROLE OF SELF INSTRUCTIONAL MATERIAL IN DISTANCE LEARNING

The need to plan effective instruction is imperative for a successful distance teaching repertoire. This is due to the fact that the instructional designer, the tutor, the author (s) and the student are often separated by distance and may never meet in person. This is an increasingly common scenario in distance education instruction. As much as possible, teaching by distance should stimulate the student's intellectual involvement and contain all the necessary learning instructional activities that are capable of guiding the student through the course objectives. Therefore, the course / self-instructional material are completely equipped with everything that the syllabus prescribes.

To ensure effective instruction, a number of instructional design ideas are used and these help students to acquire knowledge, intellectual skills, motor skills and necessary attitudinal changes. In this respect, students' assessment and course evaluation are incorporated in the text.

The nature of instructional activities used in distance education self-instructional materials depends on the domain of learning that they reinforce in the text, that is, the cognitive, psychomotor and affective. These are further interpreted in the acquisition of knowledge, intellectual skills and motor skills. Students may be encouraged to gain, apply and communicate (orally or in writing) the knowledge acquired. Intellectual-skills objectives may be met by designing instructions that make use of students' prior knowledge and experiences in the discourse as the foundation on which newly acquired knowledge is built.

The provision of exercises in the form of assignments, projects and tutorial feedback is necessary. Instructional activities that teach motor skills need to be graphically demonstrated and the correct practices provided during tutorials. Instructional activities for inculcating change in attitude and behavior should create interest and demonstrate need and benefits gained by adopting the required change. Information on the adoption and procedures for practice of new attitudes may then be introduced.

Teaching and learning at a distance eliminates interactive communication cues, such as pauses, intonation and gestures, associated with the face-to-face method of teaching. This is particularly so with the exclusive use of print media. Instructional activities built into the instructional repertoire provide this missing interaction between the student and the teacher. Therefore, the use of instructional activities to affect better distance teaching is not optional, but mandatory.

Our team of successful writers and authors has tried to reduce this.

Divide and to bring this Self Instructional Material as the best teaching and communication tool. Instructional activities are varied in order to assess the different facets of the domains of learning.

Distance education teaching repertoire involves extensive use of self-instructional materials, be they print or otherwise. These materials are designed to achieve certain pre-determined learning outcomes, namely goals and objectives that are contained in an instructional plan. Since the teaching process is affected over a distance, there is need to ensure that students actively participate in their learning by performing specific tasks that help them to understand the relevant concepts. Therefore, a set of exercises is built into the teaching repertoire in order to link what students and tutors do in the framework of the course outline. These could be in the form of students' assignments, a research project or a science practical exercise. Examples of instructional activities in distance education are too numerous to list. Instructional activities, when used in this context, help to motivate students, guide and measure students' performance (continuous assessment)



PREFACE

We have put in lots of hard work to make this book as user-friendly as possible, but we have not sacrificed quality. Experts were involved in preparing the materials. However, concepts are explained in easy language for you. We have included many tables and examples for easy understanding.

We sincerely hope this book will help you in every way you expect.

All the best for your studies from our team!



OBJECT ORIENTED ANALYSIS AND DESIGN

Contents

BLOCK 1: OBJECT ORIENTED MODELLING

UNIT 1 INTRODUCTION TO OBJECT ORIENTED MODELLING

Object Oriented Modeling, Characteristics Object Oriented Modeling: Class and Objects, Links and Association, Generalization and Inheritance, An Object Model. Benefits of OO Modeling, Introduction to OOAD tools,

UNIT 2 ADVANCE MODELING CONCEPTS

Aggregation, Abstract Class, Multiple Inheritance, Generalization as an Extension, Generalization as a Restriction, Metadata, Constraints, An Object Model

BLOCK 2: OBJECT ORIENTED ANALYSIS AND SYSTEM DESIGN

UNIT 1 OBJECT ORIENTED ANALYSIS

Object Oriented Analysis, Problem Statement: an Example, Differences between Structured Analysis and Object Oriented Analysis, Analysis Techniques: Object Modeling, Dynamic Modeling, and Functional Modeling. Adding Operations. Analysis Iteration

UNIT 2 SYSTEM DESIGN

System Design: An Object Oriented Approach, Breaking into Subsystems, Concurrency Identification, Management of data store, Controlling events between Objects, Handling Boundary Conditions

BLOCK 3: OBJECT DESIGN AND DYNAMIC MODELING

UNIT 1 OBJECT DESIGN

Object Design for Processing, Object Design Steps, Designing a Solution, Choosing Algorithms, Choosing Data Structures, Defining Classes and delegation of Responsibilities to Methods

UNIT 2 DYNAMIC MODELING

Events, State and State Diagram, Elements of State Diagrams, Examples of State Diagrams, Advance Concepts in Dynamic Modeling, Concurrency, A Dynamic model

BLOCK 4: FUNCTIONAL MODELING AND UML

UNIT 1 FUNCTIONAL MODELING

Functional Models, Data Flow Diagrams, Features of a DFD, Design flaws in DFD, A Functional model, Relationship between Object, Dynamic, and Functional Models

UNIT 2 USING UML

UML: Introduction, Object Model Notations: Basic Concepts, Structural Diagrams: Class, Object, Composite, Package, Component, Deployment. Behavioral Diagrams: Use Case, Communication, Sequence, Interaction Overview, Activity, State. Modeling with Objects

UNIT 3 CASE STUDY

This unit will cover all the OOAD aspects Covered in previous units of this course.



Dr. Babasaheb
Ambedkar
Open University

PGDCA-204

OBJECT ORIENTED ANALYSIS AND DESIGN

BLOCK 3: OBJECT DESIGN AND DYNAMIC MODELING

UNIT 1	
OBJECT DESIGN	03
UNIT 2	
DYNAMIC MODELING	11

BLOCK 3: OBJECT DESIGN AND DYNAMIC MODELING

Block Introduction

Object design is said to finding an object in object oriented analysis phases that are combined into certain classes and are refined so as to get suited for real time implementation. Dynamic modelling is said to numerous components of system having good dynamic behaviour. It shows with State diagrams wherever every state diagram represents single class with important dynamic behaviour whereas it shows interaction among classes.

In this block, we are going to detail regarding the essential of Object design with process and steps. The block can focus on the study and idea of advance ideas in Dynamic Modelling with elements related to state diagram. You can provide an inspiration on Concurrency beside dynamic modelling features.

In this block, you can make to find out and understand regarding process classes and delegation of responsibilities concerned in methods. The concept associated with solution and algorithm utilized in object designed is well explained to the students. You are going to be demonstrated practically about object designing principles and related technique.

Block Objective

After learning this block, you will be able to understand:

- About Object Design involving Processing
- Basic steps involved in Object Design
- Selecting an Object Algorithms
- Idea about working on Data Structures
- Knowledge about State Diagram
- Principles of State Diagrams with related examples

Object
Design and
Dynamic
Modeling

Block Structure

Unit 1: Object Design

Unit 2: Dynamic Modeling

UNIT 1: OBJECT DESIGN

Unit Structure

- 1.0 Learning Objectives
- 1.1 Introduction
- 1.2 Object Design for Processing
- 1.3 Object Design Steps
- 1.4 Designing a Solution
- 1.5 Choosing Algorithms
- 1.6 Choosing Data Structures
- 1.7 Defining Classes and delegation of Responsibilities to Methods
- 1.8 Let Us Sum Up
- 1.9 Answers for Check Your Progress
- 1.10 Glossary
- 1.11 Assignment
- 1.12 Activities
- 1.13 Case Study
- 1.14 Further Readings

1.0 Learning Objectives

After learning this unit, you will be able to understand:

- About Object Design
- The process involves in Object Modeling
- About Object Algorithms

1.1 Introduction

Design is the process of modelling the problem domain objects into programming constructs. Object oriented design simplifies the design process by

maintaining a one-to-one mapping between problem domain objects and software objects. To achieve object oriented design, keep your design as close as potential to problem domain objects. The interactions between your objects should mirror interactions between corresponding problem domain objects. Problem domain objects is basically an object which will be found within the problem itself.

1.2 Object Design for Processing

Object design is a design strategy where system designers assume in terms of ‘things’ rather than operations or functions. Instead of a program being designed as a set of functions that interchange data through their parameters and through a shared memory. An object-oriented program is created from interacting objects. Objects maintain their own local state and define operations on that state information.

An object-oriented design process involves designing object classes and therefore the relationships between these classes. Once the design is implemented, the desired objects are created dynamically using these class definitions.

Check your progress 1

1. The object design includes _____.
 - a. Designing system architecture
 - b. Construction of design models
 - c. Specification of object interfaces
 - d. All of these

1.3 Object Design Steps

An object contains the encapsulated data and procedures sorted along to represent an entity. The 'object interface' defines however the object will be interacted with. The planning idea relates to:

- Defining objects, making class diagram from conceptual diagram: sometimes map entity to class.
- Identifying attributes.

- Use design patterns: A design pattern isn't a finished design, it's a description of a solution to a common problem, in an exceedingly context. The main advantage of employing a design pattern is that it will be reused in multiple applications. It can also be thought of as a template for the way to resolve a problem that may be used in many different situations and/or applications. Object-oriented design patterns typically show relationships and interactions between classes or objects, without specifying the final application classes or objects that are involved.
- Define application framework: Application framework could be a term usually won't to refer to a set of libraries or classes that are wont to implement the standard structure of an application for a specific package. By bundling a large amount of reusable code into a framework, a lot of time is saved for the developer, since he/she is saved the task of rewriting large amounts of standard code for every new application that's developed.
- Identify persistent objects/data (if applicable): Identify objects that need to last longer than one runtime of the application. If a relational database is used, design the object relation mapping.
- Identify and define remote objects (if applicable).
- An object-oriented program is delineated by the interaction of these objects. Object-oriented design is that the discipline of defining the objects and their interactions to resolve a problem that was known and documented throughout object-oriented analysis.

There are certain steps concerned in designing of an object which are:

- Combining OM, FM and DM for analysis to seek out implementation categories
- Designing of algorithm to implement operation
- Optimization of access path to data
- Implementing control for external interaction
- Adjusting the class structure to increase inheritance
- Design of association
- Determine object illustration

Check your progress 2

1. While object designing which of the following should be used for analysis?
 - a. Object modeling
 - b. Functional modeling
 - c. Data modeling
 - d. All of these

1.4 Designing a Solution

To design an object we can use defining the context of a system followed by designing the architecture of the system.

- **Context:** The context of a system has two parts- static and a dynamic . The static context is designed by using a simple block diagram of the whole system which can be expanded into a hierarchy of subsystem, while the dynamic context describes how the system interacts with its environment. It is modelled by using use case diagrams.
- **System Architecture:** The system architecture is designed on the basis of the context of the system according to the principles of architectural design as well as domain knowledge.

Check your progress 3

1. How can we design an object solution?
 - a. by using context of the system
 - b. By using System architecture
 - c. Both of these
 - d. none of these

1.5 Choosing Algorithms

It is seen that in an individual Computer, there can be many operations at a particular time; hence the management is required on all running processes.

The operations within the objects are defined using algorithms. An algorithm is a stepwise procedure that solves the problem. Algorithms target that how it is to be done.

Here may be more than one algorithm for given operation. Once the alternative algorithms are known, the optimum or best suited algorithm is chosen for the given problem domain. The metrics for selecting the optimal algorithm are:

- Computational complexity: Computational complexity determines the potency of an algorithm in terms of time and memory necessities.
- Flexibility: Flexibility determines whether or not the chosen algorithm may be implemented suitably, without loss of appropriateness in varied environments.
- Understandability: This determines whether or not the chosen algorithm is easy to know and implement efficiently.

Check your progress 4

1. Which of the following metrics determine algorithm in terms of time and memory necessities?
 - a. Flexibility
 - b. Computational complexity
 - c. Understandability
 - d. None of these

1.6 Choosing Data Structures

It is seen that in a personal computer, there is several operations at a specific time; thus the management is required on all running processes that while designing data structures allow efficient algorithms.

- Many of these data structures are instances of container classes.
- Data structures include arrays, lists, queues, stacks, sets, bags, dictionaries, trees, and lots of variations, like priority queues and binary trees.

For example - ATM example.

- A transaction should have an ordered list of Updates
- By thinking about algorithms and working through the logic of an application, flaws is found that will improve a class model

Check your progress 5

1. Which of the following data structures can be used in object designing?
 - a. array
 - b. List
 - c. Queues
 - d. all of these

1.7 Defining Classes and delegation of Responsibilities to Methods

In object designing, classes and objects are grouped into packages to enable multiple groups to work together on a project. The different aspects of packaging are:

- Hiding Internal Information from Outside View
- Coherence of Elements
- Construction of Physical Modules
-

Check your progress 6

1. While designing an object_____.
 - a. Classes and objects are encapsulated in a package
 - b. Classes and objects are treated as a same thing
 - c. Both classes and objects are treated as a different thing
 - d. None of these

1.8 Let Us Sum Up

In this unit we have learnt that designing is modelling process which handles problem domain objects in programming structure and simplifies design process through one-to-one mapping among problem domain objects and software objects. It is seen that object design is design strategy where system designers think in terms of things instead of program having set of functions.

It is seen that in an individual computer, there appears many operations at particular time thereby requiring management on running processes. It is found that algorithm is a stepwise procedure that solves the problem and target of getting work done through process.

1.9 Answers for Check Your Progress

Check your progress 1

Answers: (1 –d)

Check your progress 2

Answers: (1 –d)

Check your progress 3

Answers: (1 –c)

Check your progress 4

Answers: (1 –b)

Check your progress 5

Answers: (1 –d)

Check your progress 6

Answers: (1 –a)

1.10 Glossary

1. **Design** - In modelling, problem related to object domain in programming.
2. **Object oriented design** - It is the design process which works with one-to-one mapping among problem domain objects and software objects.
3. **Object design** - It is design strategy where system designers think of operations or functions.
4. **Algorithm** - It is a stepwise procedure which solves problem in required manner.

1.11 Assignment

Explain the various object designing steps?

1.12 Activities

Study about the type of algorithm used in object designing.

1.13 Case Study

Study about the types of data types used in object design.

1.14 Further Readings

1. Design Patterns: Elements of Reusable Object-Oriented Software. Addison-Wesley. 1995.
2. O.M. Nierstrasz, A Survey of Object-Oriented Concepts.
3. In W. Kim, F. Lochovsky (eds.): Object-Oriented Concepts.
4. Databases and Applications, ACM Press and Addison-Wesley, 1989.

UNIT 2: DYNAMIC MODELING

Unit Structure

- 2.0 Learning Objectives**
- 2.1 Introduction**
- 2.2 Events**
- 2.3 State and State Diagram**
- 2.4 Elements of State Diagrams**
- 2.5 Examples of State Diagrams**
- 2.6 Advance Concepts in Dynamic Modeling**
- 2.7 Concurrency**
- 2.8 A Dynamic model**
- 2.9 Let Us Sum Up**
- 2.10 Answers for Check Your Progress**
- 2.11 Glossary**
- 2.12 Assignment**
- 2.13 Activities**
- 2.14 Case Study**
- 2.15 Further Readings**

2.0 Learning Objectives

After learning this unit, you will be able to understand:

- About working on Data Structures
- About State Diagram
- About examples related to State Diagrams

2.1 Introduction

The dynamic model basically represents the time-based aspects of a system. This is related with the temporal changes in the states of the objects in a system. The main concepts are:

- State, is the situation at a particular condition during the lifetime of an object
- Transition, a change in the state
- Event, an occurrence that triggers transitions
- Action, an uninterrupted and atomic computation that occurs due to some event
- Concurrency of transitions

A state machine models the behaviour of an object because it passes through a number of states in its lifetime due to some events as well as the actions occurring due to the events. A state machine is graphically represented by using a state transition diagram.

2.2 Events

Events are some occurrences that may trigger state transition of an object or a group of objects. Events have a location in time and space however doesn't have a period of time related to it. Events are usually related to some actions.

Examples of events are click, key press, an interrupt, stack overflow, etc. Events that trigger transitions are written alongside the arc of transition in state diagrams.

Types of Events:

External and Internal Events: External events are those events that pass from a user of the system to the objects inside the system. for example, click or key press by the user are external events. Internal events are those who pass from one object to a different object within a system. for example, stack overflow, a divide error, etc.

Deferred Events: deferred events are those that aren't immediately handled by the object in the current state however are lined up in a very queue so that they'll be handled by the object in another state at a later time.

Event Classes: Event class indicates a group of events with common structure and behaviour. Like classes of objects, event classes might also be organized in a data structure. Event classes could have attributes related to them, time being an implicit attribute.

Check your progress 1

1. What are deferred events?
 - a. Those events that pass from a user of the system to the objects inside the system
 - b. Those that pass from one object to a different object within a system.
 - c. Events are those that aren't immediately handled by the object in the current state
 - d. A group of events with common structure and behavior

2.3 State and State Diagram

A state generally represents

- a time period during which a predicate is true, e.g., budget – expenses >0,
- an action that is being performed, e.g., check inventory for order items
- or someone waits for an event to happen, e.g., arrival of a missing order item
- A state can be marked as “on” or “off”

When a state is “on”, all its outgoing transitions are eligible to fire. For a transition to fire, its event must occur and its condition must be holding true.

When a transition does fire, its action is carried out and States can have associated actions:

- On Entry-Such actions are performed soon after state gets entered
- Do-Such actions performs during lifetime of state
- On Event-Such actions performs in response to event
- On Exit-Such actions occur before state exists

- Include- Such shows submachine from another state chart diagram

State Diagrams appears to be a dynamic model which describes various states by which single object will pass during its life in response to events, along with its responses and actions.

There are certain features of State Diagrams:

- State diagrams describe all the potential states that a specific object will get into and how the object's state changes as results of events that reach the object.
- Usually state diagrams are drawn for a single class to point out the life behavior of a single object.
- An event could be a significant or noteworthy occurrence.
- A state is that the condition of an object at a moment in time, the time between events.
- A transition could be a relationship between 2 states that indicates that once an event happens the item moves from the previous state to the next state.
- A UML state diagram illustrates the interesting events and states of an object and also the behavior of an object in reaction to an event.
- It is more common to include an initial pseudo-state that automatically transitions to a different state once the instance is created.
- A state diagram can shows the lifecycle of an object: what event it experiences, its transitions and also the states it's in between these events.
- A state diagram is also applied to a variety of UML elements as well as, classes (conceptual or software) and use cases.
- A useful application of state diagrams is to explain the legal sequence of external system events that are recognized and handled by a system within the context of a use case.
- A use case state chart diagram could be a state diagram that depicts the system events and their sequence within a use case.
- State Diagrams is also created for virtually any kind or class or use case.
- If an object perpetually responds the same way to an event then it's considered state-dependent with respect to that event.

- If for all events of interest an object always reacts the same way, it's a state-independent object, in contrast state-dependent objects react otherwise to events depending on their state. State diagrams must be created for state-dependent objects with complex behavior like Use Cases, Stateful session, systems, windows, controllers, transactions, devices and role mutators.
- External Event also known as a system event is caused by something outside our system boundary.
- Internal Event is caused by something inside our system boundary.
- Temporal event is caused by the occurrence of a specific date and time or passage of time.
- Prefer using state chart diagrams to illustrate External and Temporal events (no Internal) and the reaction to them, rather than using them to design object behaviors based on internal events.
- A Transition Action is a transition that can cause an action to fire.
- A Transition Guard Condition is a transition that may also have a conditional guard or Boolean test. The transition only occurs if the test passes.
- Nested States is a sub state. A state allows nesting to contain sub states, a sub state inherits the transitions of its super state.
- Actions are associated with transitions and are considered to be processes that occur quickly and are not interrupted.
- Activities are associated with states and can take longer. An activity may be interrupted by some event.
- When a transition has no event within its label it means that the transition occurs as soon as any activity associated with the given state is completed.
- A guard is a logical condition that will return only true or false.
- A guarded transition occurs only if the guard resolves to true.
- Only one transition can be taken out of a given state, so we intend the guards to be mutually exclusive for any event.
- A super state encloses all the sub states in a single state box and defines its name in a little square at the top.
- The sub states simply inherit any transitions on the super state.

- If a state responds to an event this can be put in text in the form eventName/actionName in the state box.
- When an event is generated after a period of time this is indicated with the keyword "after", for instance you may say after 20 minutes.
- An event can be generated when a condition becomes true. You show this with the keyword "when", for instance you could have when (temperature>100 degrees).
- There are two special events "entry" and "exit". Any action that is marked as linked to the entry event is executed whenever the given state is entered via transition. The action associated with the exit event is executed whenever the state is left via transition.
- If there is a transition that goes back to the same state with an action the exit action would be executed first, then the transition's action and finally the entry action. If the state has an associated activity as well that activity is executed after the entry action.
- Concurrent state diagrams are useful when a given object has sets of independent behaviors.
- State Diagrams are good at describing the behavior of an object across several use cases. State diagrams are not very good at describing behavior that involves a number of objects collaborating.
- The syntax for a transition label has three parts all of which are optional: Event [Guard] / Action
- Activities are shown inside the class name section of the state and follows the syntax: do/activity
- States are shown in rounded rectangles.
- Concurrent state diagrams are illustrated by enclosing two state diagrams within a state box. The state box is shown by a dashed line at the middle.
- A state diagram doesn't show object interactions or collaborations.

Check your progress 2

1. What is State Diagram?
 - a. It describes all the potential states that a specific object will get into.
 - b. It is used to describe the behavior of an object across several use cases
 - c. It shows the lifecycle of an object: what event it experiences and its transitions.
 - d. All of these

2.4 Elements of State Diagrams

In order to check an interrupt obtained from device drivers, the data is assigned to the hardware which will work correctly if the data is less. The elements of state diagrams are:

- Pseudo state – is the starting point of a state chart
- State – is the condition that occurs during an object’s life when it satisfies some criteria, performs some action, or waits for an event
- Transition – is the movement of object from one state to another state
- Message event – is the trigger for the transition

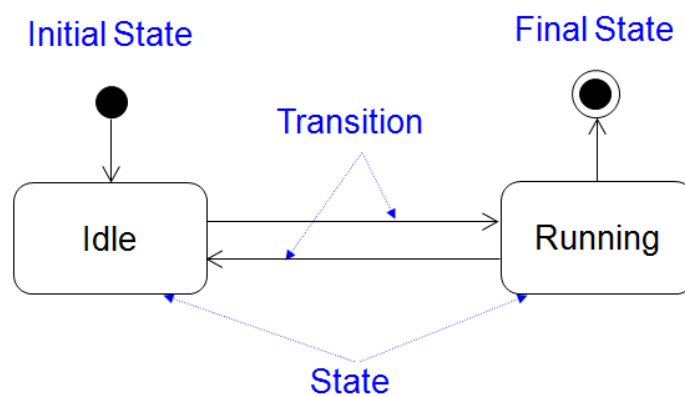


Fig 2.1 State Diagram

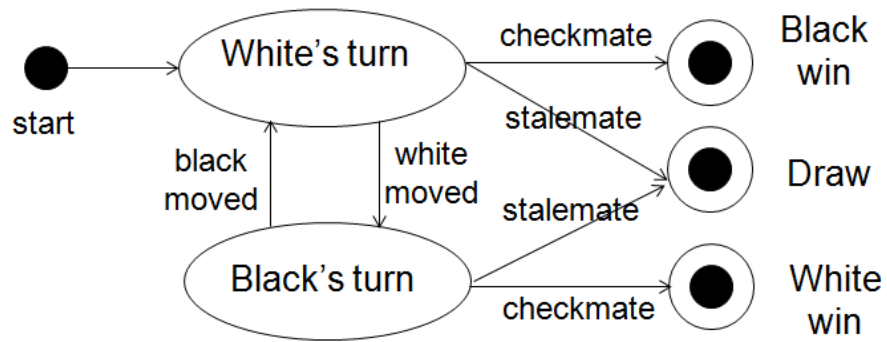


Fig 2.2 Initial and final State

Check your progress 3

1. What is pseudo state?
 - a. It is the first state
 - b. It is the starting point of a state chart
 - c. It is the final state
 - d. None of these

2.5 Examples of State Diagrams

Some of the examples where state diagrams are used are:

For book class where use of entry and exit takes place:

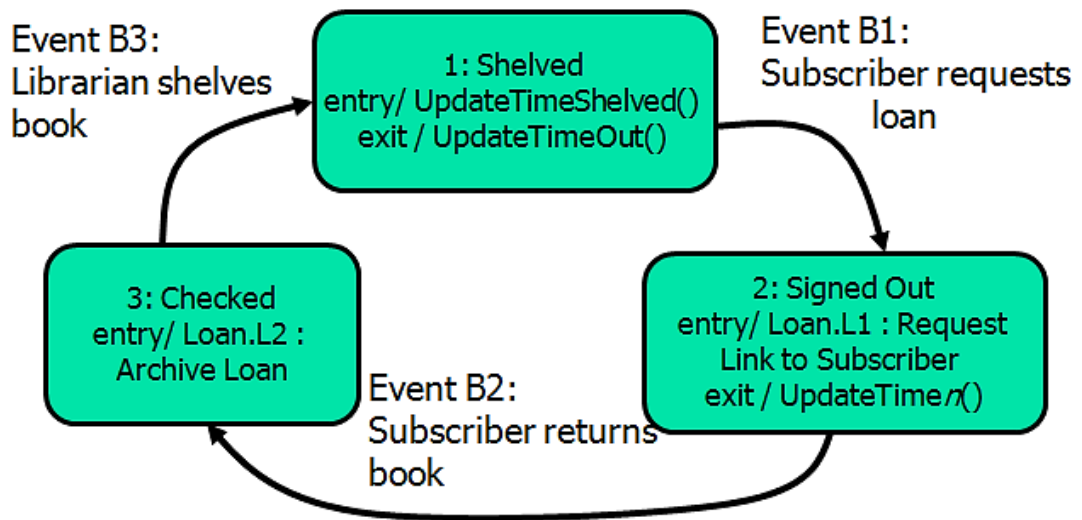


Fig 2.3 State diagram for book class

For showing life cycle of loan:

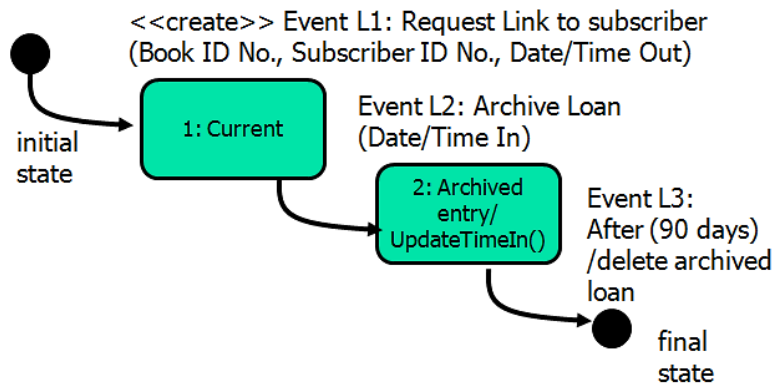


Fig 2.4 State diagram of life cycle of loan

Check your progress 4

1. State machine diagram is similar to_____.

 - a. state diagram
 - b. flowchart
 - c. data box
 - d. operation

2.6 Advance Concepts in Dynamic Modelling

Following are the 4 advanced dynamic modelling concepts:

Entry and exit actions:

Actions can be related to getting into an exciting a state as an alternative to connecting them to transactions. An entry action is performed when any transition enters the state and an exit action is performed once a state is exited. This enables a state to be expressed in terms of matched entry and exit actions without regard to what happens before a state becomes active.

An internal action doesn't change the state it executes at intervals the state. Automatic transactions fire and change the state once their conditions are met and any activity in the current state is terminated.

Sending Events:

An object can send an event {to another to a different} object together with an attribute. A race condition occurs once a state might accept events from more than one object. In this case the order of the events becomes important since it'd have an effect on the final state of the object.

Synchronization of concurrent activities:

Sometime the object might perform 2 or additional activities at a time at the same time. The internal steps might not be synchronous, but both the activities should be completed before the object will progress to next state. Any transition into a state with sub diagrams activates each of the diagrams.

Check your progress 5

1. When the exit action does is performed?
 - a. When a transaction enters the state
 - b. When the transaction leaves the state
 - c. When the task is performed
 - d. None of these

2.7 Concurrency

Concurrency is the tendency for things to happen at the same time in a system. Concurrency is a natural phenomenon, of course. Within the globe, at any given time, many things are happening at the same time. After we design software to observe and control real-world systems, we have a tendency to should deal with this natural concurrency.

When coping with concurrency problems in software systems, there are usually 2 aspects that are important: having the ability to detect and respond to external events occurring in a random order, and ensuring that these events are more established in some minimum required interval.

If every concurrent activity evolved independently, in a actually parallel fashion, this would} be comparatively simple: we have a tendency to could merely create separate programs to deal with every activity. The challenges of designing concurrent systems arise mostly because of the interactions that happen

between concurrent activities. Once co-occurring activities interact, some form of coordination is needed. An object packages procedures and data structures into a cohesive component with its own state and behaviour. It encapsulates the specific implementation of that state and behaviour and defines an interface by that other objects or software could act with it. Objects usually model globe entities or ideas, and act with other objects by exchanging messages. They're now well accepted by several because the best way to construct complex systems. Concurrency will take 2 forms in such an object model. Inter-object concurrency results once 2 or additional objects are performing activities severally via separate threads of management. Intra-object concurrency arises once multiple threads of management are active in a single object.

Two forms of concurrency:

1. System concurrency
 - The overall system is modelled because the aggregation of state diagrams
 - Each state diagram is executing concurrently with the others.
2. Concurrency inside an object
 - An object will issue co-occurring events
 - Two problems:
 - Show however control is split
 - Show how to synchronize once moving to a state without object concurrency

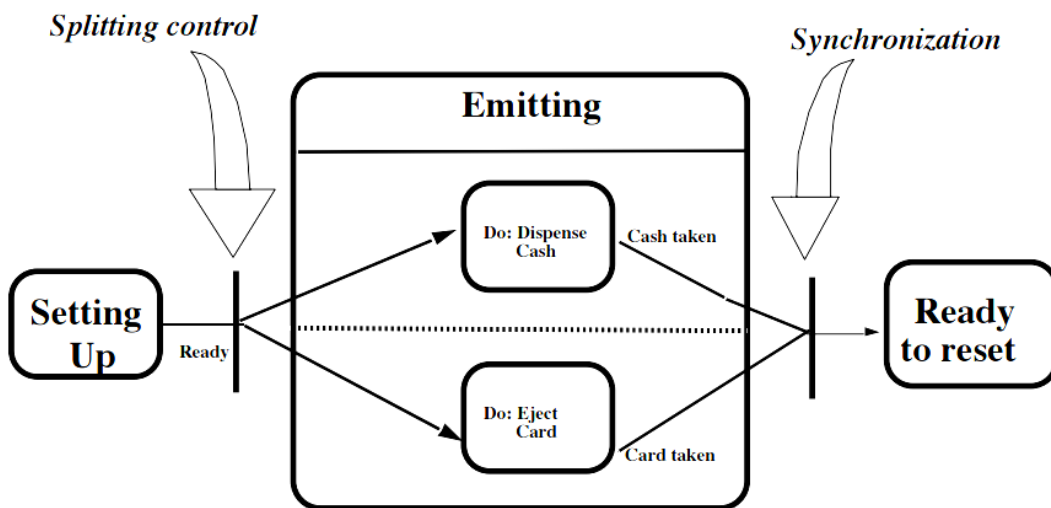


Fig 2.3 Forms of Concurrency

Check your progress 6

1. What are the forms of concurrency?
 - a. System concurrency
 - b. Concurrency within object
 - c. Both of these
 - d. None of these

2.8 A Dynamic model

A dynamic model describes about the time in which any operation is performed. It is often seen that actors are active objects where the dynamic model declares when to start or stop. Since the data stores are passive objects, so it will only react to updates and queries, hence nothing more needs to specify in case of dynamic model. We see that a dynamic model shows required change sequence to various objects in object modelling.

Check your progress 7

1. What is dynamic model?
 - a. It describes about the time of performing an event
 - b. It describes how the operations are to be done.
 - c. Both of these
 - d. None of these

2.9 Let Us Sum Up

While studying this unit, we have learnt that events are occurrences which trigger state transition of object or group that carries location in time and space without period of time related to it. It is noted that state diagrams are dynamic model showing various states by which single object which passes during its life in response to events, along with its responses and actions.

It is found that to check an interrupt from device drivers, data gets assigned to hardware which will work correctly if the data is less. It is found that concurrency appears to be tendency for things which happens at same time in a system where many things happen at same time. It is noted that dynamic model shows time in which an operation is performed where actors are active objects where dynamic model declares when to start or stop.

2.10 Answers for Check Your Progress

Check your progress 1

Answers: (1 -c)

Check your progress 2

Answers: (1 -d)

Check your progress 3

Answers: (1 -b)

Check your progress 4

Answers: (1 -a)

Check your progress 5

Answers: (1 -b)

Check your progress 6

Answers: (1 -c)

Check your progress 7

Answers: (1 -a)

2.11 Glossary

1. **Design** - In modelling, problem related to object domain in programming.
2. **Object design** - It is design strategy where system designers think of operations or functions.
3. **Algorithm** - It is a stepwise procedure which solves problem in required manner.
4. **State** - In modelling, state refers to time period during which predicate is true.

2.12 Assignment

What is the purpose of state diagrams?

2.13 Activities

What are concurrent and nested state diagrams?

2.14 Case Study

Develop state diagrams for specific classes in a given case.

2.15 Further Readings

1. Design Patterns: Elements of Reusable Object-Oriented Software. Addison-Wesley. 1995.
2. O.M. Nierstrasz, A Survey of Object-Oriented Concepts.
3. In W. Kim, F. Lochovsky (eds.): Object-Oriented Concepts.
4. Databases and Applications, ACM Press and Addison-Wesley, 1989.

Block Summary

In this block, you have learnt and understand about the basic of Object Design processing and steps. The block gives an idea on the study and concept of operations in objects through algorithms. You have been well explained on the concepts of state diagrams and basic concept on dynamic modeling.

The block detailed about the basic of various class definition along with responsibilities to object methodology. The concept related to events and its features in dynamic modelling are also explained you. You will be demonstrated practically about various elements of state diagrams.

Block Assignment

Short Answer Questions

1. What is the process involved in object modeling?
2. Explain the role of event in dynamic modeling?
3. Write note on State Diagrams?
4. Write short note on Concurrency?

Long Answer Questions

1. Write short notes on classes involved in dynamic modeling?
2. Write short note on certain examples related to State Diagrams?
3. Write note on advance concepts applied in Dynamic Modeling?

Enrolment No.

1. How many hours did you need for studying the units?

Unit No	1	2	3	4
Nos of Hrs				

2. Please give your reactions to the following items based on your reading of the block:

Items	Excellent	Very Good	Good	Poor	Give specific example if any
Presentation Quality	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	_____ _____
Language and Style	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	_____ _____
Illustration used (Diagram, tables etc)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	_____ _____
Conceptual Clarity	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	_____ _____
Check your progress Quest	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	_____ _____
Feed back to CYP Question	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	_____ _____

3. Any Other Comments

.....

.....

.....

.....

.....

.....

.....

.....



“

*Education is something
which ought to be
brought within
the reach of every one.*

”

- Dr. B. R. Ambedkar



Dr. Babasaheb Ambedkar Open University
'Jyotirmay Parisar', Opp. Shri Balaji Temple, Sarkhej-Gandhinagar Highway, Chharodi,
Ahmedabad-382 481.

OBJECT ORIENTED ANALYSIS AND DESIGN

PGDCA -204

**BLOCK 4:
FUNCTIONAL MODELING
AND UML**

**Dr. Babasaheb Ambedkar Open University
Ahmedabad**



OBJECT ORIENTED ANALYSIS AND DESIGN



Knowledge Management and
Research Organization
Pune



Editorial Panel

Author

PROF. K J Sharma

Language Editor

Mr. Vipul Shelke

Graphic and Creative Panel

Ms. K. Jamdal

Ms. Lata Dawange

Mr. Prashant Tikone

Copyright © 2015 Knowledge Management and Research Organization.

All rights reserved. No part of this book may be reproduced, transmitted or utilized in any form or by means of, electronic or mechanical, including photocopying, recording or by any information storage or retrieval system without written permission from us.

Acknowledgment

Every attempt has been made to trace the copyright holders of material reproduced in this book. Should an infringement have occurred, we apologize for the same and will be pleased to make necessary correction/amendment in future edition of this book.

The content is developed by taking reference of online and print publications that are mentioned in Bibliography. The content developed represents the breadth of research excellence in this multidisciplinary academic field. Some of the information, illustrations and examples are taken "as is" and as available in the references mentioned in Bibliography for academic purpose and better understanding by learner.'



ROLE OF SELF INSTRUCTIONAL MATERIAL IN DISTANCE LEARNING

The need to plan effective instruction is imperative for a successful distance teaching repertoire. This is due to the fact that the instructional designer, the tutor, the author (s) and the student are often separated by distance and may never meet in person. This is an increasingly common scenario in distance education instruction. As much as possible, teaching by distance should stimulate the student's intellectual involvement and contain all the necessary learning instructional activities that are capable of guiding the student through the course objectives. Therefore, the course / self-instructional material are completely equipped with everything that the syllabus prescribes.

To ensure effective instruction, a number of instructional design ideas are used and these help students to acquire knowledge, intellectual skills, motor skills and necessary attitudinal changes. In this respect, students' assessment and course evaluation are incorporated in the text.

The nature of instructional activities used in distance education self-instructional materials depends on the domain of learning that they reinforce in the text, that is, the cognitive, psychomotor and affective. These are further interpreted in the acquisition of knowledge, intellectual skills and motor skills. Students may be encouraged to gain, apply and communicate (orally or in writing) the knowledge acquired. Intellectual-skills objectives may be met by designing instructions that make use of students' prior knowledge and experiences in the discourse as the foundation on which newly acquired knowledge is built.

The provision of exercises in the form of assignments, projects and tutorial feedback is necessary. Instructional activities that teach motor skills need to be graphically demonstrated and the correct practices provided during tutorials. Instructional activities for inculcating change in attitude and behavior should create interest and demonstrate need and benefits gained by adopting the required change. Information on the adoption and procedures for practice of new attitudes may then be introduced.

Teaching and learning at a distance eliminates interactive communication cues, such as pauses, intonation and gestures, associated with the face-to-face method of teaching. This is particularly so with the exclusive use of print media. Instructional activities built into the instructional repertoire provide this missing interaction between the student and the teacher. Therefore, the use of instructional activities to affect better distance teaching is not optional, but mandatory.

Our team of successful writers and authors has tried to reduce this.

Divide and to bring this Self Instructional Material as the best teaching and communication tool. Instructional activities are varied in order to assess the different facets of the domains of learning.

Distance education teaching repertoire involves extensive use of self-instructional materials, be they print or otherwise. These materials are designed to achieve certain pre-determined learning outcomes, namely goals and objectives that are contained in an instructional plan. Since the teaching process is affected over a distance, there is need to ensure that students actively participate in their learning by performing specific tasks that help them to understand the relevant concepts. Therefore, a set of exercises is built into the teaching repertoire in order to link what students and tutors do in the framework of the course outline. These could be in the form of students' assignments, a research project or a science practical exercise. Examples of instructional activities in distance education are too numerous to list. Instructional activities, when used in this context, help to motivate students, guide and measure students' performance (continuous assessment)



PREFACE

We have put in lots of hard work to make this book as user-friendly as possible, but we have not sacrificed quality. Experts were involved in preparing the materials. However, concepts are explained in easy language for you. We have included many tables and examples for easy understanding.

We sincerely hope this book will help you in every way you expect.

All the best for your studies from our team!



OBJECT ORIENTED ANALYSIS AND DESIGN

Contents

BLOCK 1: OBJECT ORIENTED MODELLING

UNIT 1 INTRODUCTION TO OBJECT ORIENTED MODELLING

Object Oriented Modeling, Characteristics Object Oriented Modeling: Class and Objects, Links and Association, Generalization and Inheritance, An Object Model. Benefits of OO Modeling, Introduction to OOAD tools,

UNIT 2 ADVANCE MODELING CONCEPTS

Aggregation, Abstract Class, Multiple Inheritance, Generalization as an Extension, Generalization as a Restriction, Metadata, Constraints, An Object Model

BLOCK 2: OBJECT ORIENTED ANALYSIS AND SYSTEM DESIGN

UNIT 1 OBJECT ORIENTED ANALYSIS

Object Oriented Analysis, Problem Statement: an Example, Differences between Structured Analysis and Object Oriented Analysis, Analysis Techniques: Object Modeling, Dynamic Modeling, and Functional Modeling. Adding Operations. Analysis Iteration

UNIT 2 SYSTEM DESIGN

System Design: An Object Oriented Approach, Breaking into Subsystems, Concurrency Identification, Management of data store, Controlling events between Objects, Handling Boundary Conditions



BLOCK 3: OBJECT DESIGN AND DYNAMIC MODELING

UNIT 1 OBJECT DESIGN

Object Design for Processing, Object Design Steps, Designing a Solution, Choosing Algorithms, Choosing Data Structures, Defining Classes and delegation of Responsibilities to Methods

UNIT 2 DYNAMIC MODELING

Events, State and State Diagram, Elements of State Diagrams, Examples of State Diagrams, Advance Concepts in Dynamic Modeling, Concurrency, A Dynamic model

BLOCK 4: FUNCTIONAL MODELING AND UML

UNIT 1 FUNCTIONAL MODELING

Functional Models, Data Flow Diagrams, Features of a DFD, Design flaws in DFD, A Functional model, Relationship between Object, Dynamic, and Functional Models

UNIT 2 USING UML

UML: Introduction, Object Model Notations: Basic Concepts, Structural Diagrams: Class, Object, Composite, Package, Component, Deployment. Behavioral Diagrams: Use Case, Communication, Sequence, Interaction Overview, Activity, State. Modeling with Objects

UNIT 3 CASE STUDY

This unit will cover all the OOAD aspects Covered in previous units of this course.



Dr. Babasaheb
Ambedkar
Open University

PGDCA-204

OBJECT ORIENTED ANALYSIS AND DESIGN

BLOCK 4: FUNCTIONAL MODELING AND UML

UNIT 1

FUNCTIONAL MODELING 02

UNIT 2

USING UML 15

Unit 3

CASE STUDY 36

BLOCK 4: FUNCTIONAL MODELING AND UML

Block Introduction

Functional Modelling is a methodology which is linked with object oriented analysis model that describes about function of internal processes of system by the way of Data Flow Diagrams. These diagram shows how data is worked out by system and describes flow of data or information right from data entering to data processed with idea on its storage.

In this block, we will detail about the basic characteristics of data Flow Diagrams techniques with graphical arrangement of data representing of information. The block will focus on Unified Modeling Language with study about meta models features for reactive systems are discussed. The concept of Object and dynamic model features are well explained.

In this block, the student will made to learn and understand about the basic of Interface template where various functions are implementing with class interface. The concept related to Functional Modelling and its related process is well explained to the students.

Block Objective

After learning this block, you will be able to understand:

- Features of Functional model
- Characteristics of data Flow Diagrams
- Idea about Object and dynamic model
- Features of Unified Modeling Language
- Concept of Behavioral Diagrams

Block Structure

Unit 1: Functional Modeling

Unit 2: Using UML

Unit 3: Case Study

UNIT 1: FUNCTIONAL MODELING

Unit Structure

- 1.0 Learning Objectives**
- 1.1 Introduction**
- 1.2 Functional Models**
- 1.3 Data Flow Diagrams**
- 1.4 Features of a DFD**
- 1.5 Design flaws in DFD**
- 1.6 A Functional model**
- 1.7 Relationship between Object**
- 1.8 Dynamic and Functional Models**
- 1.9 Let Us Sum Up**
- 1.10 Answers for Check Your Progress**
- 1.11 Glossary**
- 1.12 Assignment**
- 1.13 Activities**
- 1.14 Case Study**
- 1.15 Further Readings**

1.0 Learning Objectives

After learning this unit, you will be able to understand:

- The Study of Functional Model
- The Study of Data Flow Diagram
- The Study of Dynamic Model

1.1 Introduction

Functional Modelling is a tool that allows a team or an individual to produce a behavioural/operational model of an existing or planned system. The resulting model shows the system functionality and the logical interconnections between that functionality. In essence it describes how the system functionality has to cooperate to deliver the Operational Requirements of the system.

1.2 Functional Models

Functional Modelling is a process associated with object oriented analysis model which explains about function of internal processes of a system with the use of Data Flow Diagrams. It represents functional derivation of data values not stressing on their origination when calculated. It is prepared from algorithms which relates to present entities. In real world processes, it produces such model that shows goal oriented properties of modelled entity.

Functional Modelling is a way where group or an individual will be allowed to design behavioural or operational model which is based on present/current system. It describes system working and interconnections which exists among functionality. It shows how system functionality behave on order to supply operational needs of system. On framing such models, it is possible to:

- Have necessary system functionality
- Check basic operational concept
- Find logical system interfaces

There are certain advantages of Functional Modelling:

- It helps in logical working of modeling team with proposed or existing system functionality features and locates for missing features from customer needs
- Locates the possible interfaces of system and upgrade their explanation
- Favours team members in considering more and more system of interest
- Explains more understanding about system of interest and customers expectations
- Shows customer interest models
- Allow the team members to share information with proper understanding

Check your progress 1

1. What are the advantages of using functional modelling?
 - a. It shows system functionality and its interconnections
 - b. It is used to locate the possible interfaces of system
 - c. It describes the function of internal processes of a system
 - d. All of these

1.3 Data Flow Diagrams

Data flow diagram shows how data is worked out by system. It shows the flow of data or information right from data entering to data processed with idea on its storage. There are 2 type of notations used in data flow diagrams:

Process Notations: Such type of notation deals with processing of incoming data and flowing of process data.

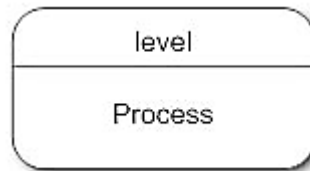


Fig 1.1 Process Notation

Data store Notations: Datastores are location of data in system. It can called as files.

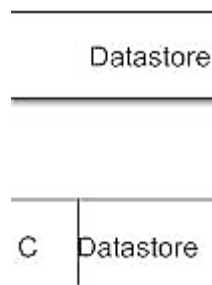


Fig 1.2 Datastore Notation

Data flow Notations: The dataflow notation represents a channel through which packets of information flows. It can be shows with the use of arrows having data names.

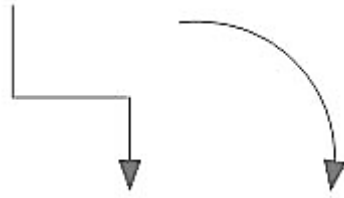


Fig 2.3 Dataflow Notation

External Entity Notations: It is a type of notation which are objects that lies outside the system which can be applied for system communication. It serves as source and destination of system's inputs and outputs.

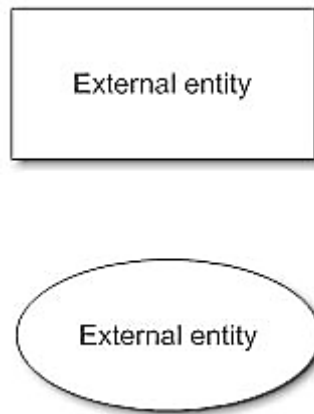
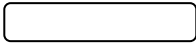
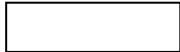



Fig 1.4 Entity Notations

Check your progress 2

1. Which notation is used to describe datastore?

- a. _____

- b. 
- c. 
- d. 

1.4 Features of a DFD

It is noted that Data flow diagram shows graphical arrangement of representing data or information which summarizes data movement from processing steps which occurs in business process. They separate the collections of data and stores the data that gets accumulated at the time of process and locate data sources that occur beyond process boundaries.

Characteristics

It has certain characteristics features such as:

Data Flow Summarisation:

It has characteristics features of summarizing data of a process on single page since the details can be shows in concise manner about a system process.

Completeness:

It explains about the completeness of process model by understanding data which is required by information system. Here the questions can be generated very fast.

Process:

It is called as processing steps activities where it shows the problems of information systems about processing of work that comes at the time of business process.

Patterns:

It serves as shortcut for understanding patterns about data flows which is designed for business processes. It highlight large amounts of data which are collected, stored, transferred, generated, used, and delivered by highlighting areas of potentially irrelevant activities.

Advantages of DFDs

- It is a strong and simple graphic technique that can be easily understandable.
- It shows information about inputs and outputs that can be easily followed by the people.
- It describes system at various levels of details
- It shows boundaries of the system.
- It can be applied in interviews.

- It locates easily about user information which is based on future requirements.

Check your progress 3

1. What are the key features of data flow diagram?
 - a. Completeness
 - b. Process
 - c. Patterns
 - d. All of these

1.5 Design flaws in DFD

Information can be easily understood by using Data Flow Diagrams and on the same time it is hard to draw the correct path of data. By not understood the data correctly, it is hard to draw DFDs and there are chances of mistakes. Normally people make certain common mistakes and sometimes misunderstood the process because of certain flaws in it.

It is easy to analyse that every process result in start and end. The common patterns of data flow involve flows which begin or end at required processing step. It is correct as probably data cannot flow or goes out of its own until processed. This highlights that there are certain mistakes which can be easily findout and rectified while drawing a DFD. The figure 1.5 shows certain symbols which are purposefully left blank as it is easy in finding patterns for drawing DFD.

Wrong	Right	Description
		A source or a sink cannot provide data to another source or sink without some processing occurring.
		Data cannot move directly from a source to a data store without being processed.
		Data cannot move directly from a data store to a sink without being processed.
		Data cannot move directly from one data store to another without being processed.

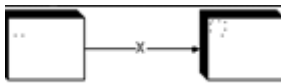
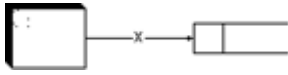

Fig 1.5 Comparisons among DFD

Apart from this, another flaw with DFD appears when outputs from single processing step mismatches with its input. This happens when:

- When processing step carries only flow which is termed as black hole situation.
- When processing step carries only output flows with no input flow, which could be a case of miracle.
- When processing step carries output flow more than the total inputs, which shows case of grey hole.

Check your progress 4

1. Which is the correct way of transferring data between source and sink?

- a. 
- b. 
- c. 
- d. None of these

1.6 A Functional model

It is normally seen that a function model is just like activity model or can be like a process model which is graphical drawing of someone function having a confined scope. The idea of function model is to explain the function and processes and helps with new information by identifying opportunities and creates a base for finding product and service costs.

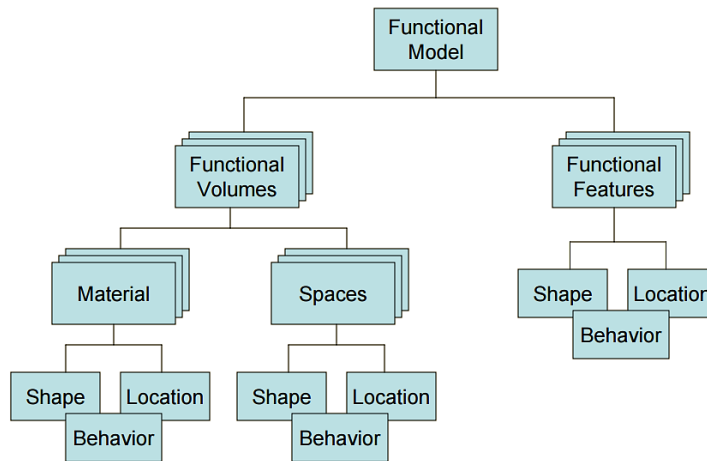


Fig 1.6 Functional Models

The functional modeling view is strictly on showing dynamic process as seen in fig 1.6. The concept behind such modeling is related to process, function, transformation, activity, action, task etc. So a good example of modeling language using such view is data flow diagrams. The functional model uses four symbols to describe a process such as:

- Process: It shows transformation from input to output.
- Store: It shows data collection or material.
- Flow: It shows flow of data or material in a particular process.
- External Entity: It is an interactive external model system.

A functional model with reference to fig 1.7 shows:

- Functions. It can be creating an order and approving an order.
- Data flows. It shows purchasing of data from purchase officer in order to generate an order function.
- External entities. It can a entity which is external to the system.
- Data stores. It shows that the orders are lying in order data store.

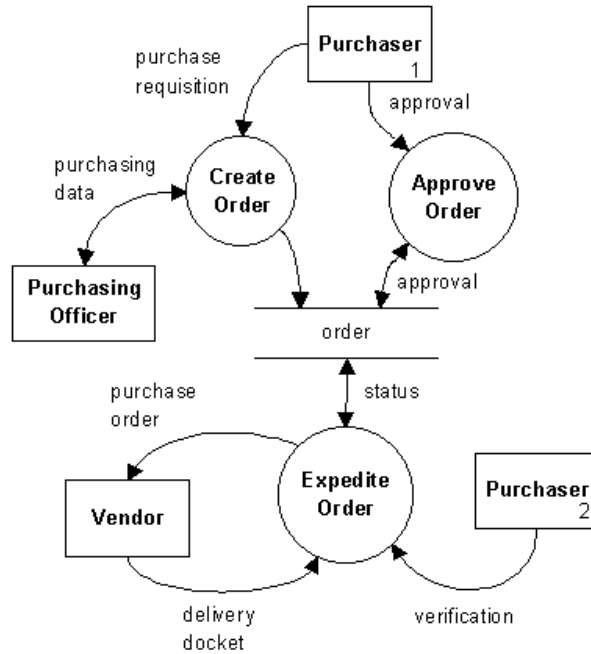


Fig 1.7 Function representation of DFD

Check your progress 5

1. What do you mean by store in functional modeling?
 - a. It is an external model system
 - b. Collection of data
 - c. A collection of functions
 - d. All of these

1.7 Relationship between Object

There occur relationships of object either in dynamic modelling or in functional modelling.

Object and dynamic model

We see that a dynamic model shows required change sequence to objects in object modeling. In this, the states of dynamic model relates with the classes of qualities and show linkage values of an object. In this, events can be show as operations on object model which serves as concepts in terms of generalization, aggregation and inheritance.

Generalization:

It is noticed that in generalization by restriction, every object is restricted to the values which an object carry. Both generalizations based on classes and on states partition will have required object values. In case of inherent, the differences among objects are modelled differently as per class. In case of temporary difference, they are modelled as different states of similar class.

Aggregation:

Aggregation occurs a combined state with more substate. It occurs at two levels which could be aggregation of objects and aggregation in objects

Inheritance:

You can draw branches of events and states. Events left with feature of their parents while states left with required transitions. It is found that the dynamic model of a class also inherited by its subclasses.

Object and functional model

It is found that every components of functional model is related to object model which are:

Processes: In functional model, processes get implemented in the object method. Processes in functional model describes about the objects which are associated with function. Over and over again, one in or output appears as client of a process, while the other inputs results as supplier. Such type of client supplier relationships starts with implementation of dependencies which occurs among required classes.

Actors: It represents as objects in object model.

Data stores: It represents as objects in object model or characteristic of an object

Data flows: It shows values in object model where data flowing to or from actors describes operations on or by objects. Also it is noted that data flowing to or from data stores also shows queries or updates.

Check your progress 6

1. What is inheritance?
 - a. Acquiring properties of base class in sub class
 - b. A method
 - c. A class
 - d. None of these

1.8 Dynamic and Functional Models

The dynamic model describes about the time of performing an operation, while functional model describes about how the operations gets done and which type of arguments are required during the operation. There is however a difference among both the operations in terms of operation on actors and operations on data stores. As actors are active objects, so dynamic model has to declare when to start or stop. Since the data stores are passive objects, so it will react to updates and queries, hence nothing needs to specify in case of dynamic model.

Check your progress 7

1. What is dynamic model?
 - a. It describes about the time of performing an event
 - b. It describes how the operation is to be done.
 - c. Both of these
 - d. None of these

1.9 Let Us Sum Up

In this unit we have learnt that Data flow diagram shows graphical arrangement of representing data or information which summarizes data movement from processing steps which occurs in business process

The idea of function model is to explain the function and processes and helps with new information by identifying opportunities and creates a base for finding product and service costs.

The dynamic model describes about the time of performing an operation, while functional model describes about how the operations gets done and which type of arguments are required during the operation.

1.10 Answers for Check Your Progress

Check your progress 1

Answers: (1 –d)

Check your progress 2

Answers: (1 –a)

Check your progress 3

Answers: (1 –d)

Check your progress 4

Answers: (1 –c)

Check your progress 5

Answers: (1 –b)

Check your progress 6

Answers: (1 –a)

Check your progress 7

Answers: (1 –a)

1.11 Glossary

1. **Object** - It is an instance of a class.
2. **Property** - It is an object characteristic.
3. **Method** - It is an object capability which is a subroutine or function associated with a class.
4. **Constructor** - It is a method called at the moment an object is instantiated.
5. **Inheritance** - It is a class which inherits characteristics from another class.
6. **Encapsulation** - It is a method of bundling data and methods which make use of data.

1.12 Assignment

Explain in detail about Data flow diagram.

1.13 Activities

Create an activity on function model.

1.14 Case Study

Is dynamic model more lucrative than functional model?

1.15 Further Readings

1. Doo-Kwon Baik eds. (2005). Systems modeling and simulation: theory and applications : third Asian Simulation Conference, AsiaSim 2004, Jeju Island, Korea, October 4–6, 2004. Springer, 2005. ISBN 3-540-24477-8.
2. Derek W. Bunn, Erik R. Larsen (1997). Systems modelling for energy policy. Wiley, 1997. ISBN 0-471-95794-1
3. Hartmut Ehrig et al. (eds.) (2005). Formal methods in software and systems modeling. Springer, 2005 ISBN 3-540-24936-2

UNIT 2: USING UML

Unit Structure

2.0 Learning Objectives

2.1 UML: Introduction

2.2 Object Model Notations: Basic Concepts

2.3 Structural Diagrams: Class, Object, Composite, Package, Component, Deployment

2.4 Behavioural Diagrams: Use Case, Communication, Sequence, Interaction Overview, Activity, State

2.5 Modeling with Objects

2.6 Let Us Sum Up

2.7 Answers for Check Your Progress

2.8 Glossary

2.9 Assignment

2.10 Activities

2.11 Case Study

2.12 Further Readings

2.0 Learning Objectives

After learning this unit, you will be able to understand:

- About UML
- About Object Model Notations
- About Structural Diagrams

2.1 UML: Introduction

One of the languages that are gaining exponential popularity and usage is the Unified Modelling Language (UML). UML is a graphical modelling language that supposedly unifies and integrates the various notations used before by the

various methodologies proposed. It is the requirement as amount of object-oriented methods increased from less to much higher. It constitutes the required standard notation and semantics for correctly describing software built with object oriented or component-based technology. It's undoubtedly a step in the right direction; however it's not an ideal or universal modelling language. we tend to believe that UML, as it stands today, must be, in some contexts and for a few application domains, complemented with alternative meta-models or a minimum of adapted to address those meta-models.

UML has 2 totally different meta-models for this purpose: state charts and activity diagrams. These 2 meta-models present several necessary characteristics for reactive systems, particularly concurrency and hierarchy, however they do not allow an elegant treatment of the data path/plant resources and also the specification of dynamic parallelism. These are 2 crucial necessities for embedded software, since totally different parts of the system may attempt to access at the same time the same resources.

UML was created mainly with the ideas and concepts from the OOSE, OMT-2, and Booch methods. OMT-2 was a revision of OMT with the formal introduction of use cases into the methodology. OMT-2 has special expressiveness for analysis and data-intensive info systems and since that special type of systems has sometimes a trivial functional model, it's simple to know the non-inclusion of DFDs among UML.

2.2 Object Model Notations: Basic Concepts

The object is represented in the same way as the class. The only difference is the name which is underlined as shown below.

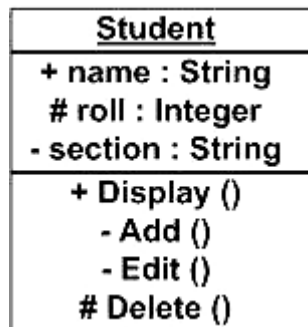


Fig 2.1 Object Notation

As object is actual implementation of a class which is known as the instance of a class. So it has the same usage as that of class.

Interface is represented by a circle as shown below. It has a name which is generally written below the circle.

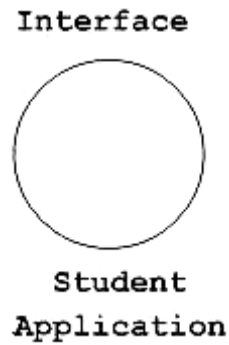


Fig 2.2 Interface Notation

Interface is used to describe the functionality without implementation. Interface is just like a template where you can define different functions and not the implementation. When a class implements the interface it also implements the functionality as per the requirement.

Collaboration is represented by a dotted ellipse as shown below. It has a name written inside the ellipse.

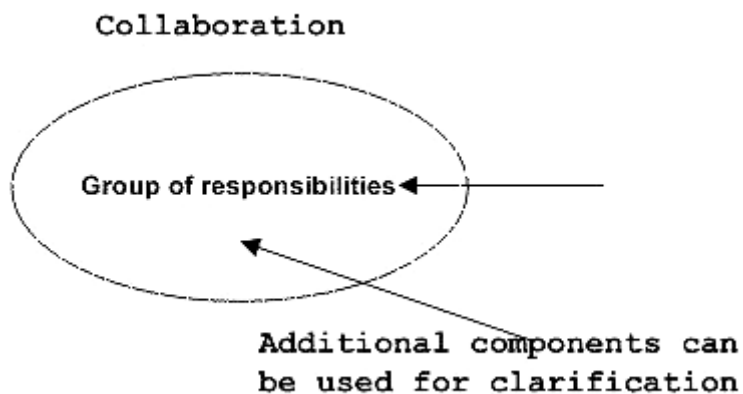


Fig 2.3 Collaboration Notation

Collaboration represents responsibilities. Generally responsibilities are in a group.

Use case is represented as an ellipse with a name inside it. It may contain additional responsibilities.

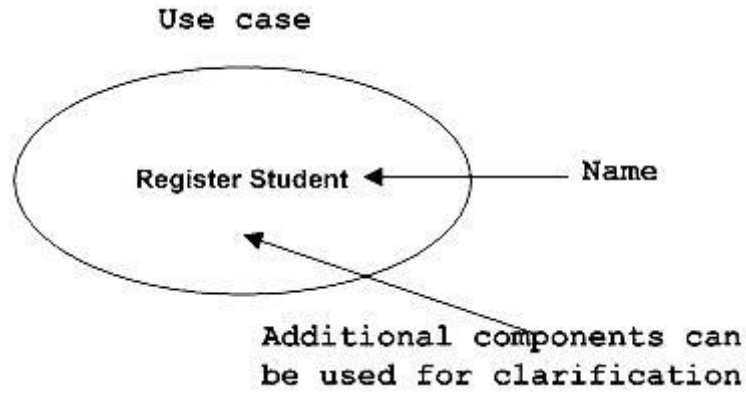


Fig 2.4 Use case Notation

Use case is used to capture high level functionalities of a system.

An actor can be defined as some internal or external entity that interacts with the system.

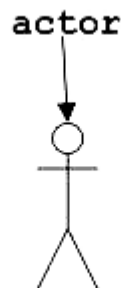


Fig 2.5 Actor Notation

Actor is used in use case diagram to describe the internal or external entities.

Initial state is defined to show the start of a process. This notation is used in almost all diagrams.

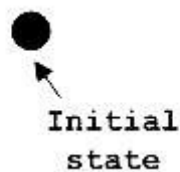


Fig 2.6 Initial state Notation

The usage of Initial State Notation is to show the starting point of a process.

Final state is used to show the end of a process. This notation is also used in almost all diagrams to describe the end.



Fig 2.7 Final state Notation

The usage of Final State Notation is to show the termination point of a process.

Active class looks similar to a class with a solid border. Active class is generally used to describe concurrent behaviour of a system.

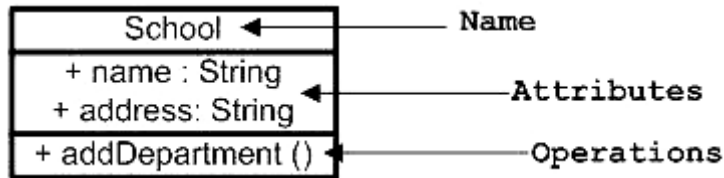


Fig 2.8 Active class Notation

Active class is used to represent concurrency in a system.

A component in UML is shown as below with a name inside. Additional elements can be added wherever required.

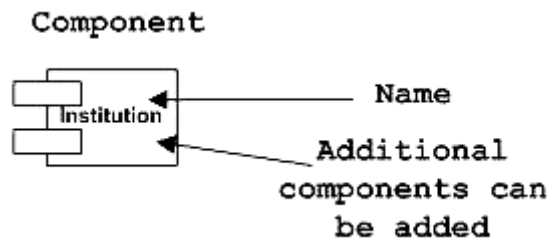


Fig 2.9 Component Notation

Component is used to represent any part of a system for which UML diagrams are made.

A node in UML is represented by a square box as shown below with a name. A node represents a physical component of the system.

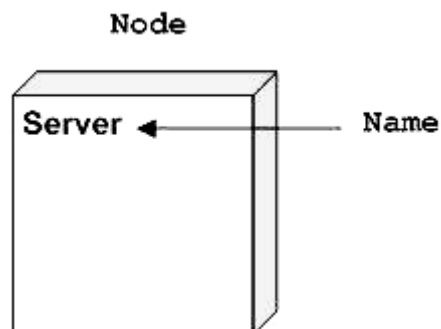


Fig 2.10 Node Notation

Node is used to represent physical part of a system like server, network etc.

Check your progress 1

1. How can we represent an interface?
 - a. By using a circle
 - b. By using an eclipse
 - c. By using a square
 - d. None of these

2.3 Structural Diagrams: Class, Object, Composite, Package, Component, Deployment

Structure diagrams show the things in a system being modelled. In a more technical term they show different objects in a system.

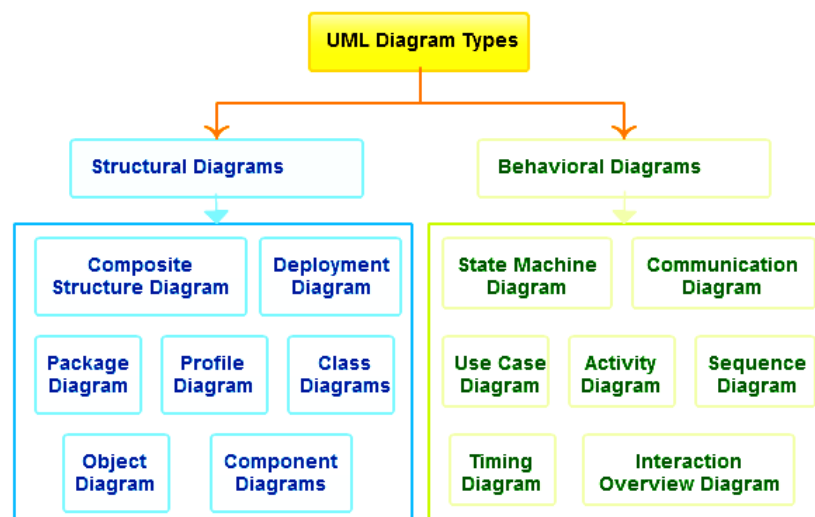


Fig 2.11 Structural Diagrams

Class Diagram

Class diagrams are arguably the foremost used UML diagram type. It's the most building block of any object oriented answer. It shows the classes in an exceedingly system, attributes and operations of {each of every} class and the relationship between each class.

- Most common diagram.

- Shows a group of classes, interfaces, and collaborations and their relationships (dependency, generalization, association and realization); notes too.
- Represents the static view of a system (With active classes, static process view)

Three modelling views for class Diagram:

- Conceptual: the diagram reflects the domain
- Specification: focus on interfaces of the software (Java supports interfaces)
- Implementation: class (logical database schema) definition to be enforced in code and database.

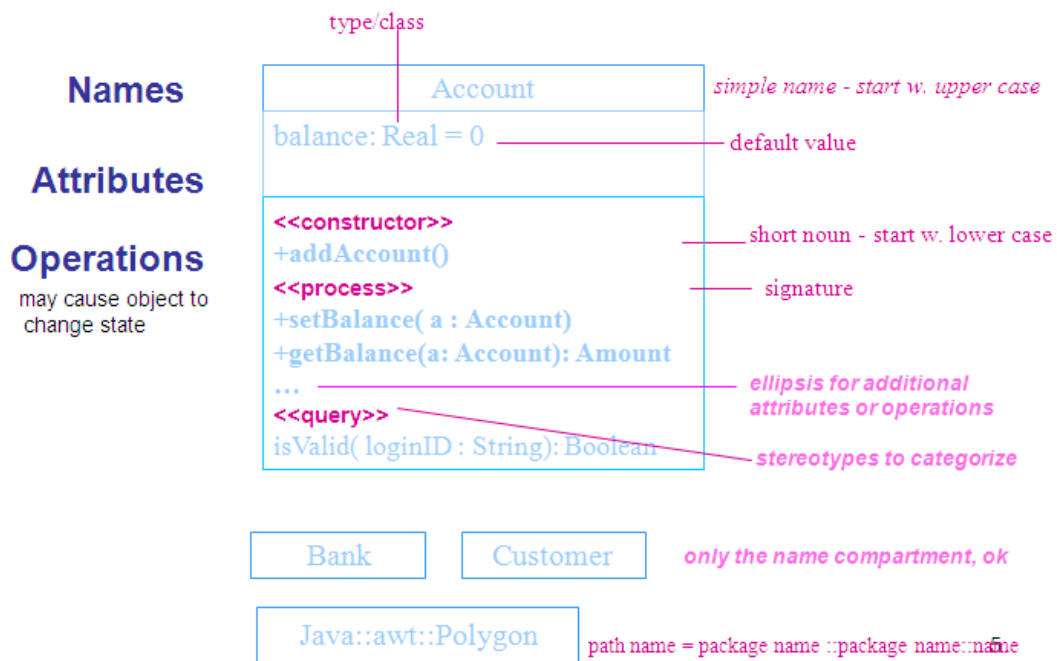


Fig 2.12 Class Diagram

Component Diagram

A component diagram displays the structural relationship of components of the software system. These are generally used when working with complex systems that have many components. Components can communicate with each other using interfaces. The interfaces are linked by using connectors. Below images shows a component diagram.

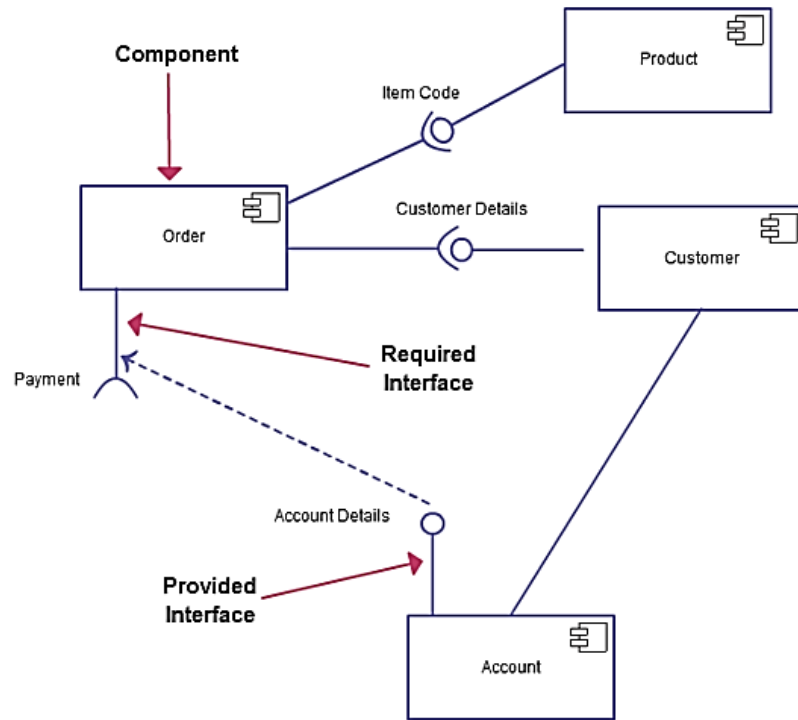


Fig 2.13 Component Diagram

Deployment Diagram

A deployment diagram describes various hardware components of system along with software. Such diagrams are required when software solution is put across many machines with each having different configuration as shown in figure:

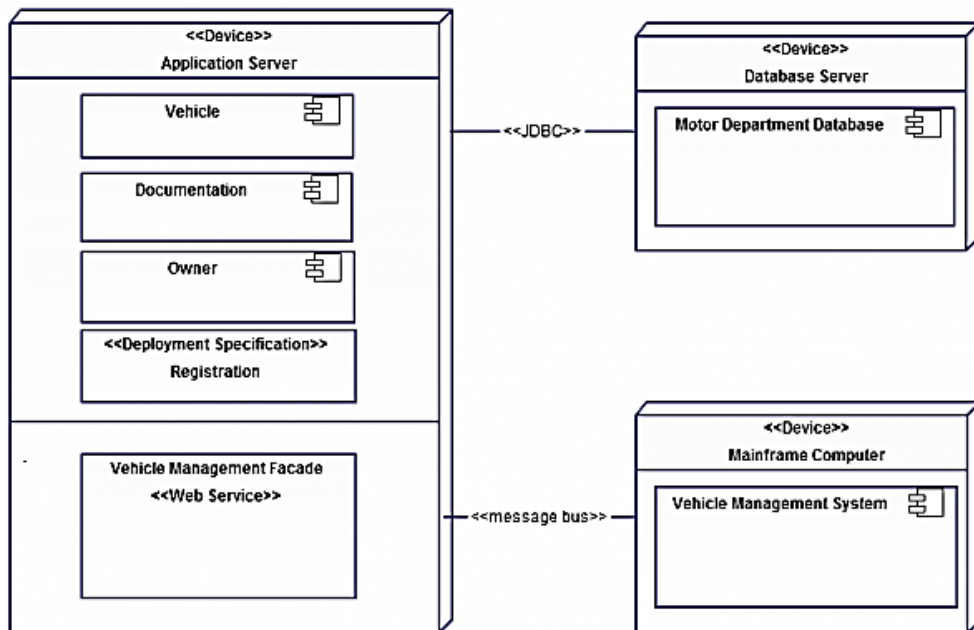


Fig 2.14 Deployment Diagram

Object Diagram

Object Diagram is an Instance diagrams which is similar to class diagrams since it describes relationship among objects used in real world. Such diagrams are applied to describe about system appearance at particular time. With support of data available in objects explains complex relationships among objects.

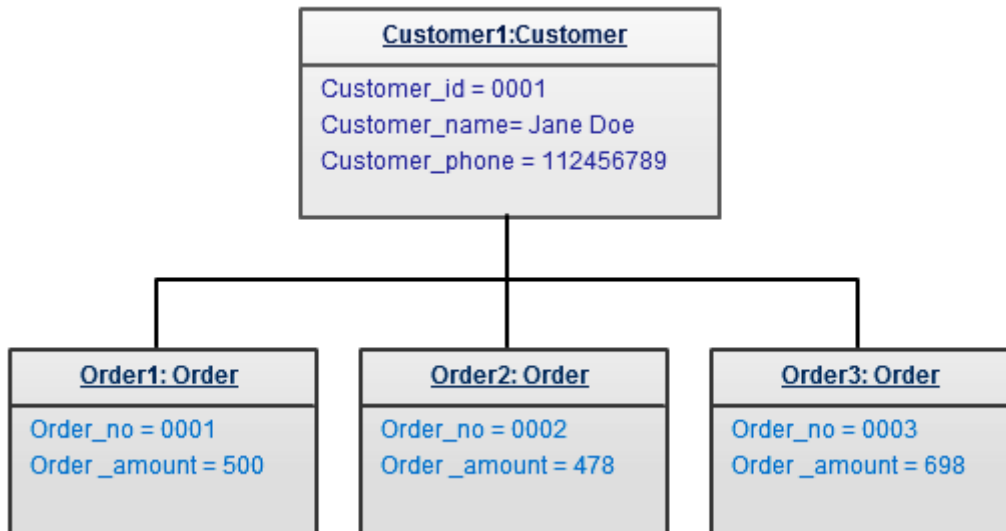


Fig 2.15 Object Diagram

Package Diagram

As the name suggests a package diagrams shows the dependencies between different packages in a system.

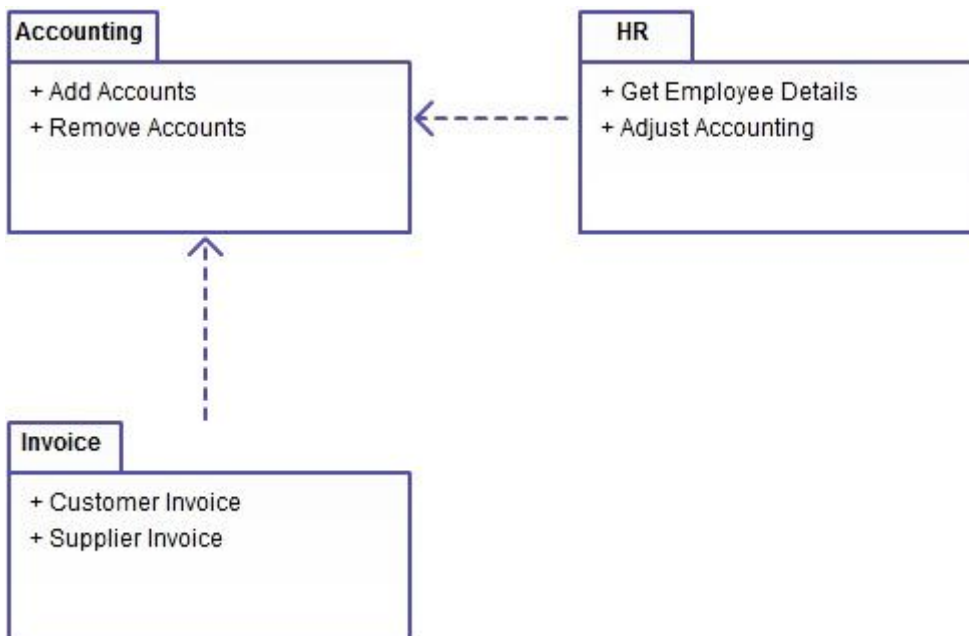


Fig 2.16 Package diagram

Composite Structure Diagram

Composite structure diagrams are used to show the internal structure of a class.

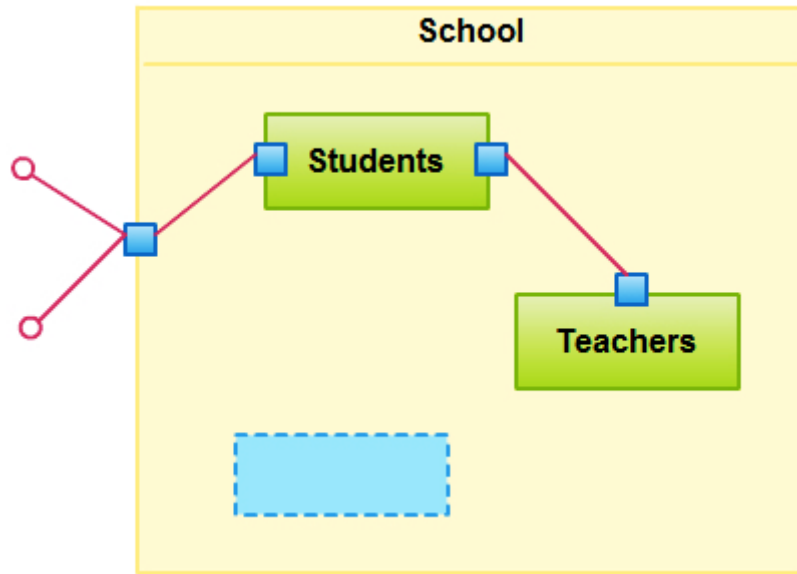


Fig 2.17 Composite Diagram

Check your progress 2

1. Which of the following diagram is used to show the hardware and software of the system?
 - a. Class diagram
 - b. Deployment diagram
 - c. Package diagram
 - d. None of these

2.4 Behavioural Diagrams: Use Case, Communication, Sequence, Interaction Overview, Activity, State

Use Case Model

The use case model captures the necessities of a system. Use cases are a means of communicating with users and different stakeholders what the system is meant to try to. A use case is a single unit of meaningful work. It provides a high-

level view of behaviour observable to somebody or something outside the system. The notation for a use case is an ellipse.

A use case usually includes:

- Name and description
- Requirements
- Constraints
- Scenarios
- Scenario diagrams
- Additional info.

Actors

It is a type of use case diagram which shows interaction with system and entities that is external to system. Such external entities called actors showing roles of human users, external hardware.

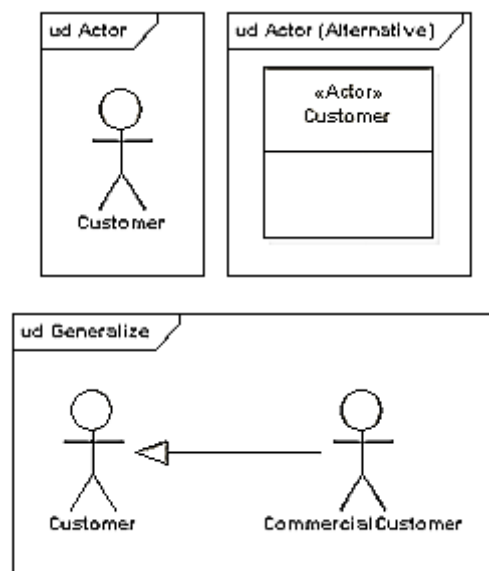


Fig 2.18 Use case diagram

Communication Diagrams

Communication diagram is another type describes interactions which occurs among elements at run-time through inter-object relationships. It is a diagram describing interaction among elements at run-time in same way as Sequence diagram. They are normally applied to judge inter-object relationships, while Sequence diagrams are good at judging processing over time.

Communication diagrams employ ordered, labelled associations to illustrate processing. Numbering is important to indicate the order and nesting of processing. A numbering scheme could be:

1

1.1

1.1.1

1.1.2

1.2 And so on.

This example shown in fig 2.19 is a communication diagram which uses message levels in order to gather related flows and various colours of messages.

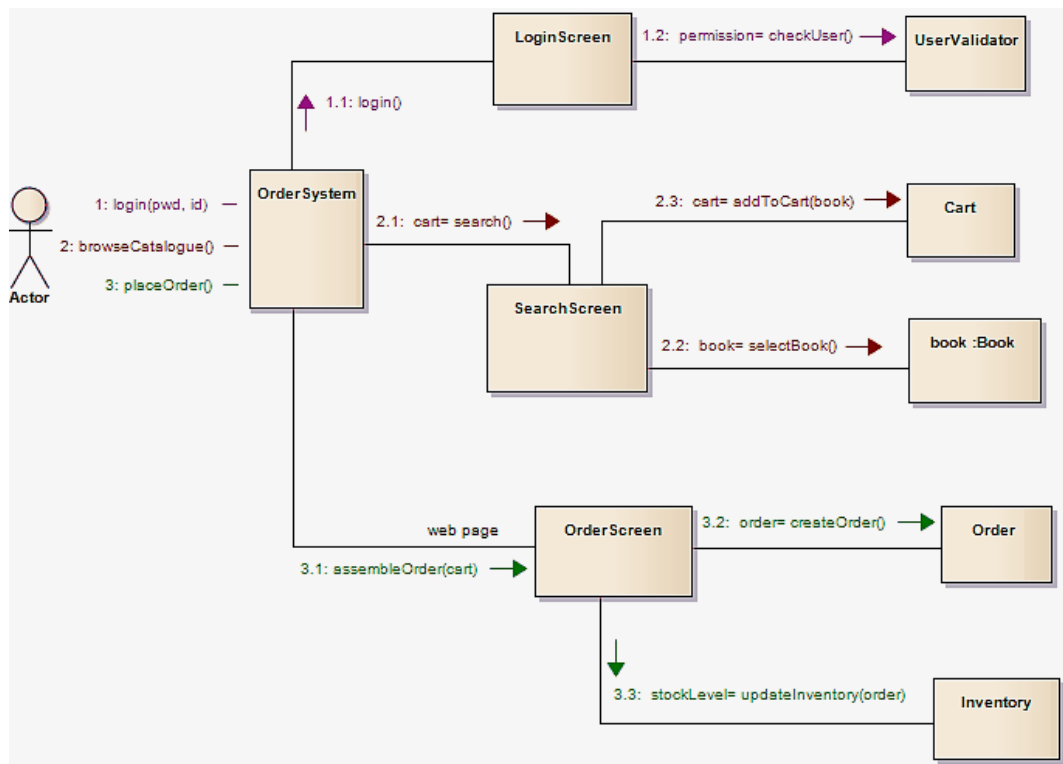


Fig 2.19 Communication Diagram

Sequence Diagrams

Sequence diagrams are structural description about behaviour showing series of sequential steps over time. They represent work flow, message transfer and shows about elements in normal cooperate over time in order to get the result.

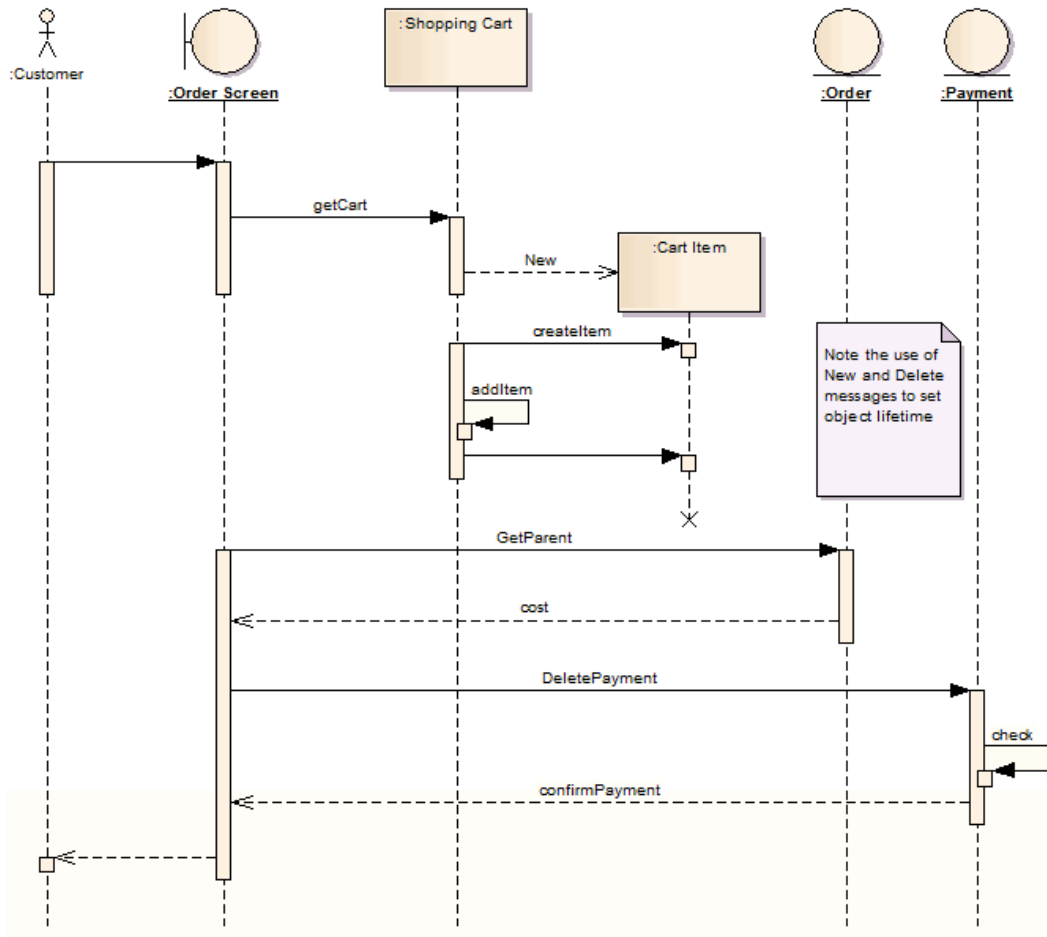


Fig 2.20 Sequence diagram

A Sequence diagram is a structured representation of behaviour as a series of sequential steps over time.

Use to

- Depict work flow, message passing and how elements normally cooperate over time to achieve a result
- Capture the flow of information and responsibility throughout the system, early in analysis; messages between elements eventually become method calls in the class model
- Make instructive models to be used Case scenarios; by creating a Sequence diagram with an Actor and parts involved within the Use Case, you'll model the sequence of steps the user and therefore the system undertake to complete the required tasks

Construction:

- Each sequence element is arranged in a very horizontal sequence, with messages passing back and forward between elements

- Messages on a Sequence diagram are of many types; the Messages may also be configured to reflect the operations and properties of the source and target elements.
- An Actor part is used to represent the user initiating the flow of events
- Stereotyped elements, like Boundary, control and Entity, is used to illustrate screens, controllers and database things, severally
- Each element has a dashed stem known as a Lifeline, wherever that element exists and potentially takes part in the interactions

Interaction overview

Interaction overview diagrams visualize the cooperation between different interaction diagrams a control flow serving an encompassing purpose.

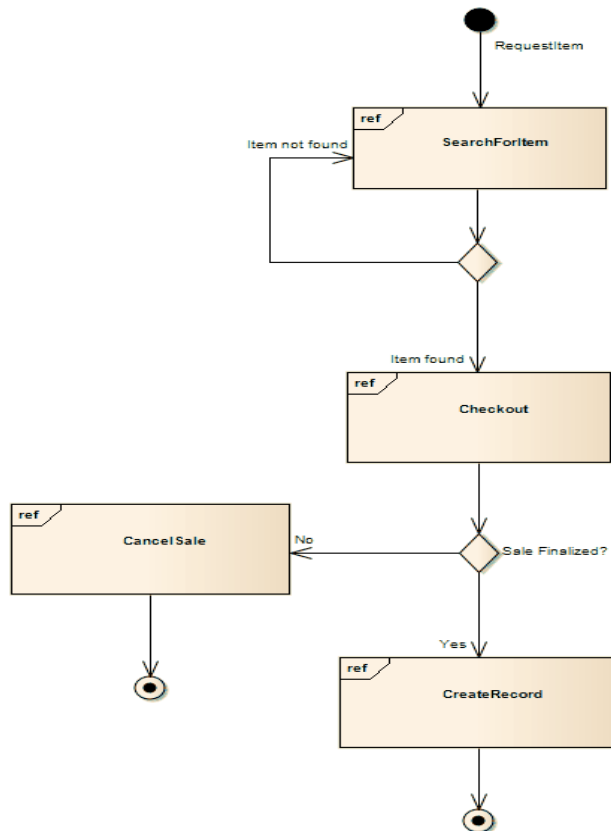


Fig 2.21 Interaction overview

Fig 2.16 shows sample sale process with sub-processes abstracted within Interaction Occurrences. It is smaller than Activity diagram, and is conceptualized in the same way where when flow moves in interaction, then the respective interaction's process follows before Interaction Overview's flow can advance.

The Interaction overview diagrams visualize the cooperation among other interaction diagrams to illustrate example as an instance parenthetically let's say

maybe} a control flow serving an encompassing purpose. As Interaction overview diagrams are a variant of Activity diagrams, most of the diagram notation is that the same, as is that the process of constructing the diagram.

Decision points, Forks, Joins, start points and end points are the same. Instead of Activity elements, however, rectangular elements are used. There are 2 types of these elements:

- Interaction elements show an inline Interaction diagram, which can be any one of the four varieties
- Interaction occurrence elements are references to an existing Interaction diagram: they're visually represented by a frame, with ref in the frame's title space; the diagram name is indicated within the frame contents.

Activity Diagrams

Activity diagrams model the behaviours of a system, and also the way in which these behaviours are connected in an overall flow of the system.

Activity diagrams are used to model the behaviours of a system, and also the means during which these behaviours are connected in an overall flow of the system. The logical paths a process follows, supported various conditions, concurrent processing, data access, interruptions and alternative logical path distinctions, are all accustomed construct a process, system or procedure.

The following diagram illustrates some of the features of Activity diagrams, including Activities, Actions, Start Nodes, End Nodes and Decision points.

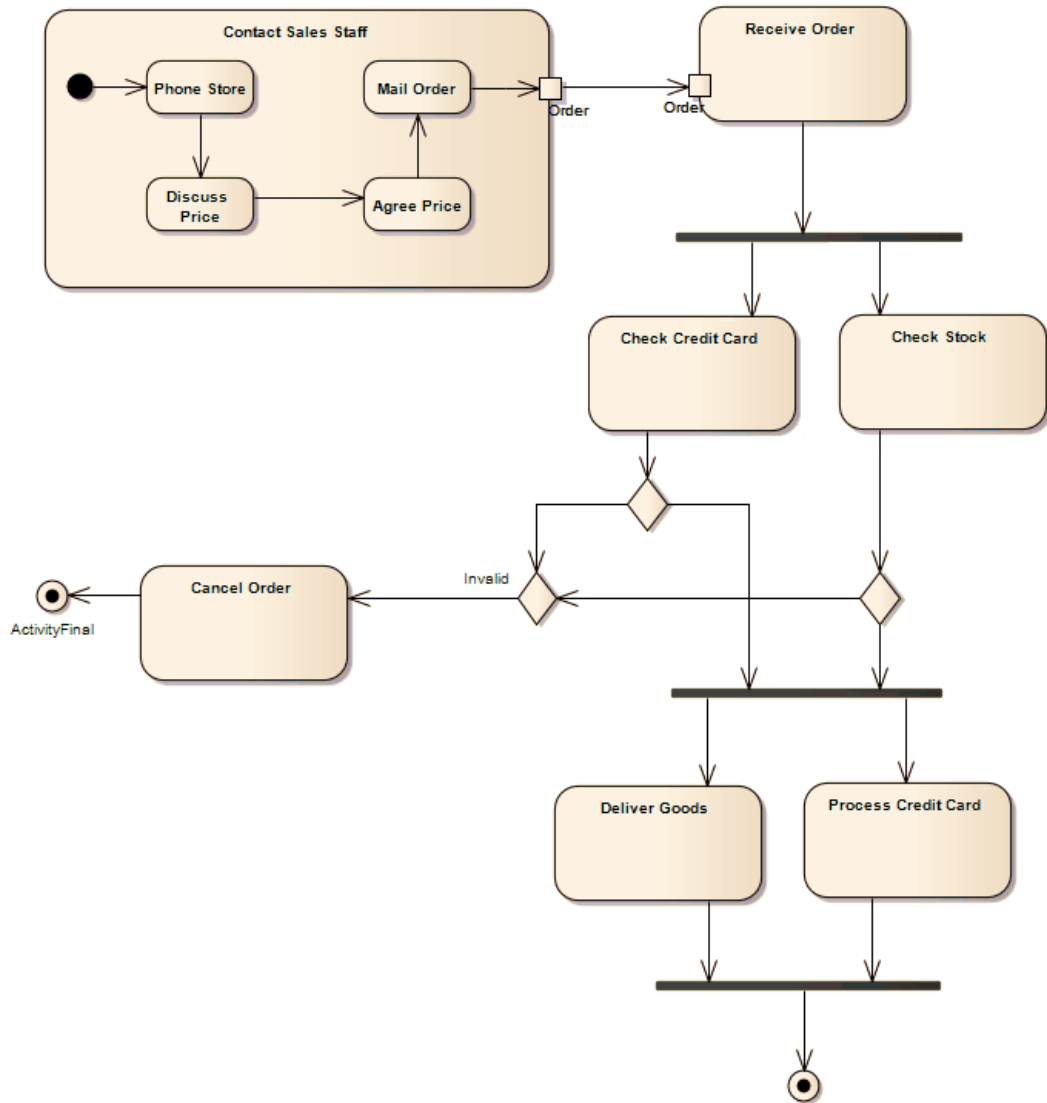


Fig 2.22 Activity Diagram

State Machine Diagrams

State Machine diagrams describes about an element that moves among states, which classifies behaviour as per transition, triggers and constraining guards. Naming conventions:

- State Machines were formerly known as State diagrams
- State Machine representations in UML are based on the Harel State Chart Notation and therefore are sometimes referred to as State Charts.

The diagram shown in fig 2.23 represents some of the features of State Diagram.

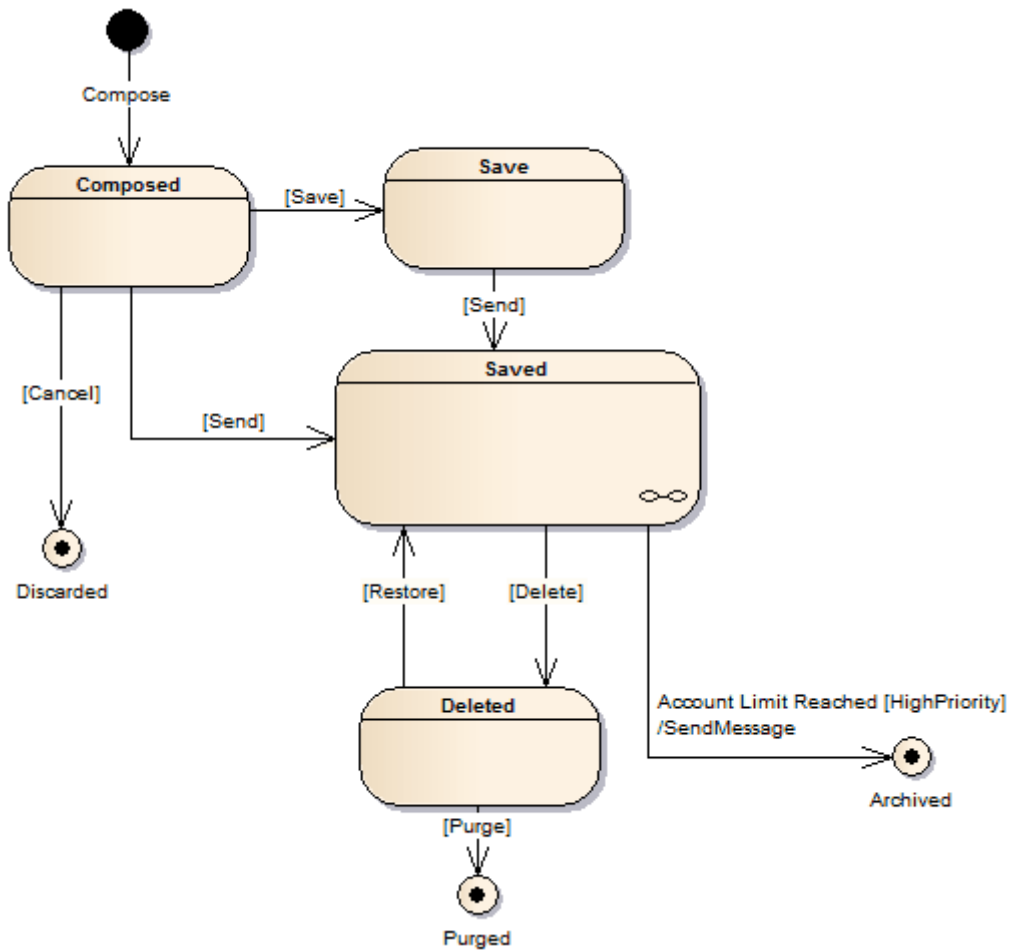


Fig 2.23 State diagram

Check your progress 3

1. Which of the following is used to describe the movements of elements between states?
 - a. State machine diagram
 - b. Deployment diagram
 - c. Sequence diagram
 - d. None of these

2.5 Modelling with Objects

Modelling with Object may be a variety of information central categories that summarize to packed things. They are:

- Very common and applied in all applications
- Model problem domain objects
- mapped to records corresponding database table
- used as return values for data Access Object methods
- tested with the help of JUnit tool
- used to implement Model in Model-View-Controller arrangements

To know more about object modelling, first we want to make UML object diagram that is drawn on board or paper. Such diagram won't carry any information associated with classes, however contains detail concerning objects and their relationships with one another. Also, it carries attributes and values.

To start with object modelling, we tend to use ludo game as real life example that carries counters, board and die along with certain rules. Fig. 2.24 highlight with modelling game where 2 players hit a counter.

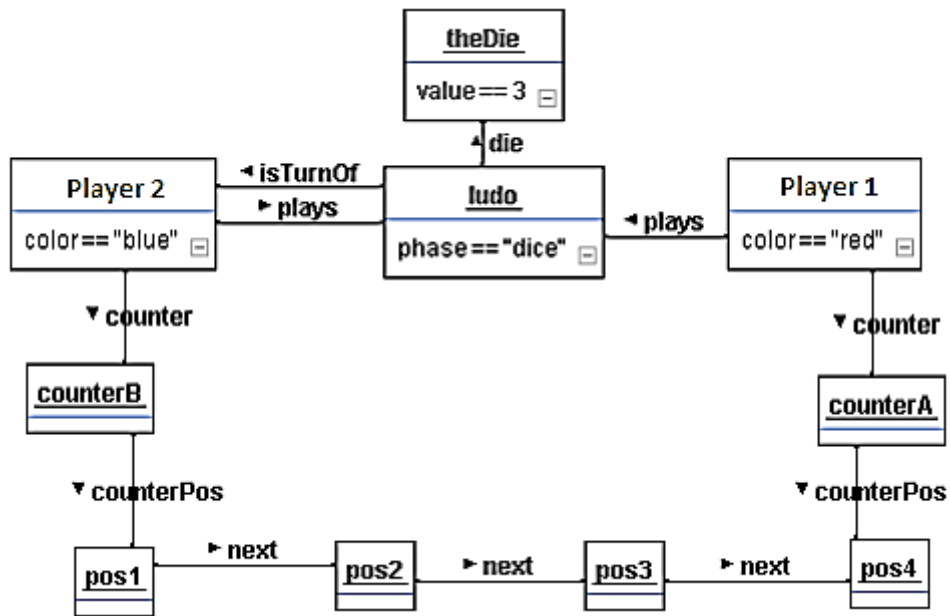


Fig 2.24 Modelling game

In fig. 2.24, player 2 and player 1 are both modelled as objects. Such objects have color feature modelling that player 1 plays with red counters and player 2 plays with blue counter. There are two counters where counter A belongs to player 1 and counter B belongs to player 2. It is modelled with two counter links. It is seen that player 2 counter is placed on field pos1 with next field as pos2 and so on. It seems that player 1 counter is placed 3 fields further of player 2 counter. Now the Die object shows as die currently shows 3.

Check your progress 4

1. What are the advantages of using object modelling?
 - a. It models the problem domain objects
 - b. It can easily mapped the records to corresponding database table
 - c. It is used as return values for data Access Object methods
 - d. All of these

2.6 Let Us Sum Up

In this unit we have learnt that UML contains 2 different meta models as state charts and activity diagrams having different characteristics for reactive systems, particularly concurrency and hierarchy. It is noted that there are important necessities for embedded software as different parts of system attempts to access at same time the same resources.

It is studied that interface describes functionality without implementation which is like a template where different functions are defined without implementation. A class diagram is an arguably foremost applied UML diagram type which serves as building block of objects oriented answer showing classes in exceedingly system, attributes and class operations.

The object diagram is Instance diagrams which is similar to class diagrams that shows relationship among objects used in real world and are applied to describe about system appearance at particular time. The activity diagrams are used to model the behaviours of a system which means during which these behaviours are connected in an overall flow of the system.

2.7 Answers for Check Your Progress

Check your progress 1

Answers: (1 –a)

Check your progress 2

Answers: (1 –b)

Check your progress 3

Answers: (1 –a)

Check your progress 4

Answers: (1 –d)

2.8 Glossary

1. **UML** - A type of meta models having different characteristics for reactive systems.
2. **Class diagram** - An arguably foremost UML diagram type serving as building block of objects oriented answer showing classes in system, attributes and class operations.
3. **Object diagram** - Similar to class diagrams showing relationship among objects in real world that describes about system appearance at particular time.
4. **Activity diagram** - The model showing behaviour of system where behaviours are connected in overall flow of system.

2.9 Assignment

Explain about deployment diagram.

2.10 Activities

Write a DSM system in C++ using MPI for the underlying message-passing and process communication.

2.11 Case Study

Discuss modelling with objects in detail.

2.12 Further Readings

1. Schulte, J. Niere: Thinking in Object Structures: Teaching Modelling in Secondary Schools; in Sixth Workshop on Pedagogies and Tools for Learning Object Oriented Concepts, ECOOP, Malaga, Spain, 2002.
2. Zündorf: Rigorous Object Oriented Software Development, Habilitation Thesis, University of Paderborn, 2001.

UNIT 3: CASE STUDY: CAR RENTAL

Unit Structure

3.0 Learning Objectives

3.1 Case Study

3.2 Let Us Sum Up

3.3 Glossary

3.4 Assignment

3.5 Activities

3.6 Case Study

3.7 Further Readings

3.0 Learning Objectives

After learning this unit, you will be able to understand:

- About Object modeling
- About Case Study model

3.1 Case Study

Object oriented technology is currently a famous technology in which the programs are implemented having structure applied as in real world modelling and interaction for reusability.

The case study is of car rental administration system where central UML diagrams and language features are highlighted along with details of use case, class, object, state chart, sequence, collaboration and activity diagrams. The case study shows a typical development process where diagram order does not reflect order which comes in development part.

The case diagram of car rental is shown in fig 3.1.

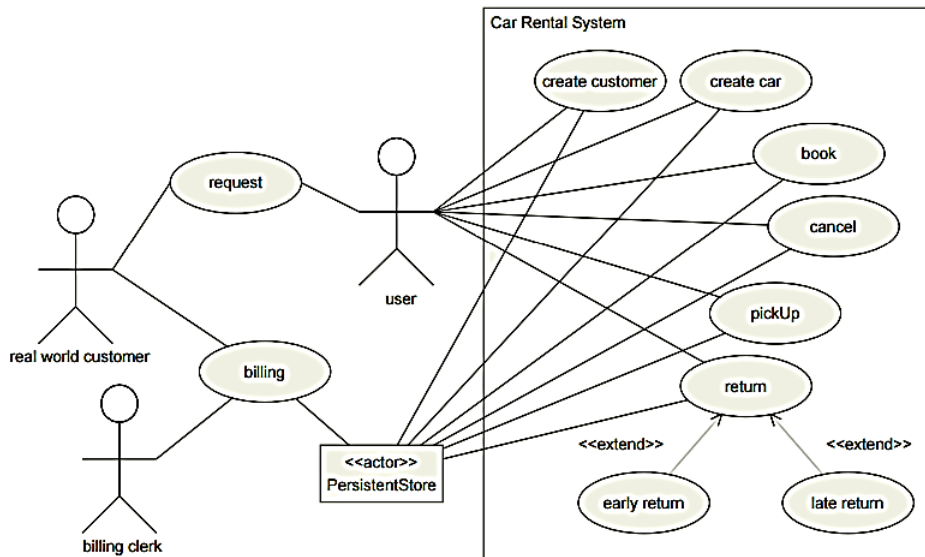


Fig 3.1 Car rental case diagram

In the above use case diagram, to find:

Details regarding create customer, we require:

- Use case name: create customer
- Goal: creating new customer
- Precondition: recording of real world customer currently absent
- Postcondition: new customer exists
- Actors: user
- Triggering event: recording of real world customer

Details regarding create car, we require:

- Use case name: creating car
- Goal: creation of new car
- Precondition: recording of real world car presently absent
- Postcondition: existence of new car
- Actors: user
- Triggering event: recording of real world car

Details regarding Use Case book

- Use case name: case book
- Goal: booking detail of car rental

- Precondition: pause of booking details
- Postcondition: open booking or new booking
- Actors: user
- Triggering event: request of real world customer

Details regarding Use Case cancel

- Use case name: cancel
- Goal: preventing car that should be taken on booking
- Precondition: current status of cancel of booking
- Postcondition: booking closed as no car is picked for booking
- Actors: user
- Triggering event: booking cancellation of real world customer requests

Details regarding Use Case pickUp

- Use case name: pickUp
- Goal: delivery of car for car rental
- Precondition: availability of booking
- Postcondition: car marked as unavailable and booking becomes current booking
- Actors: user
- Triggering event: real world customer requests pick up

Details for Use Case return

- Use case name: return
- Goal: returning car for car rental
- Precondition: delivery of car on current booking
- Postcondition: closing of booking; availability of car
- Actors: user
- Triggering event: request on end day for car booking in real world

Fig 3.2 shows allowed object diagram with 3 classes and 3 objects:

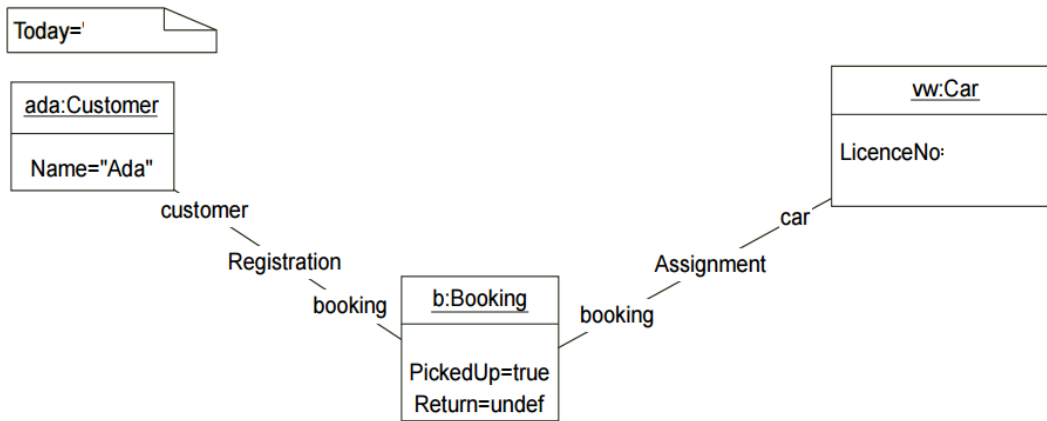


Fig 3.2 allowed object diagram

Fig 3.3 shows disallowed object diagram:

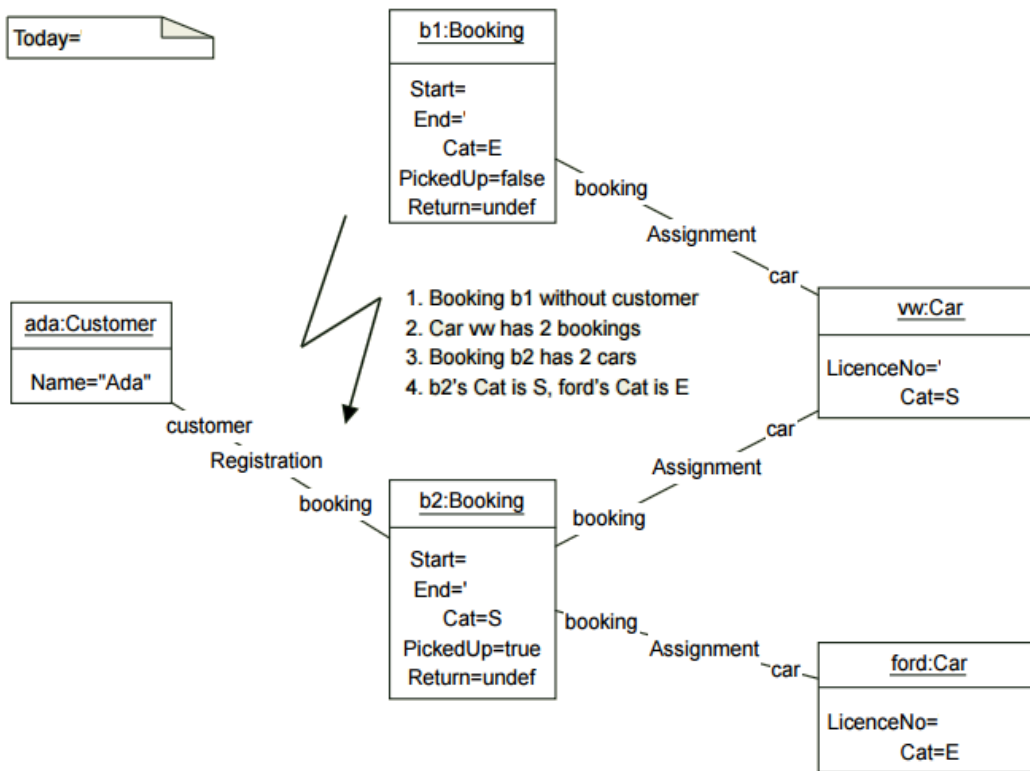


Fig 3.3 disallowed object diagram

The main idea of customer state diagram of car booking is shown in fig 3.4.

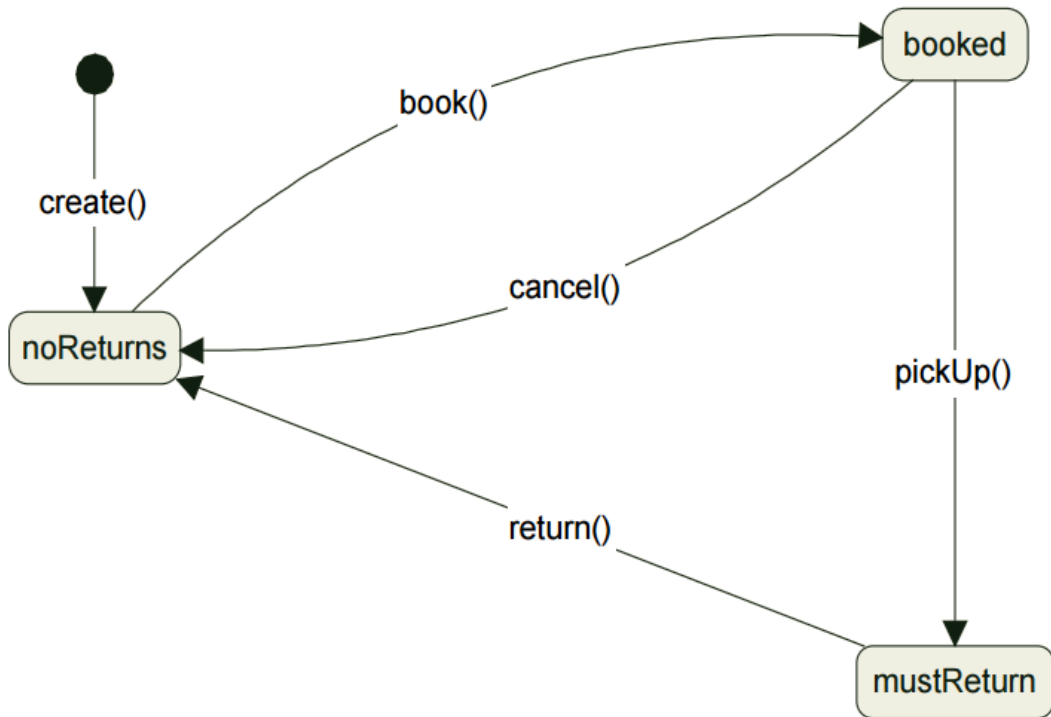


Fig 3.4 customer state diagram

Fig 3.5 shows textual details about car details:

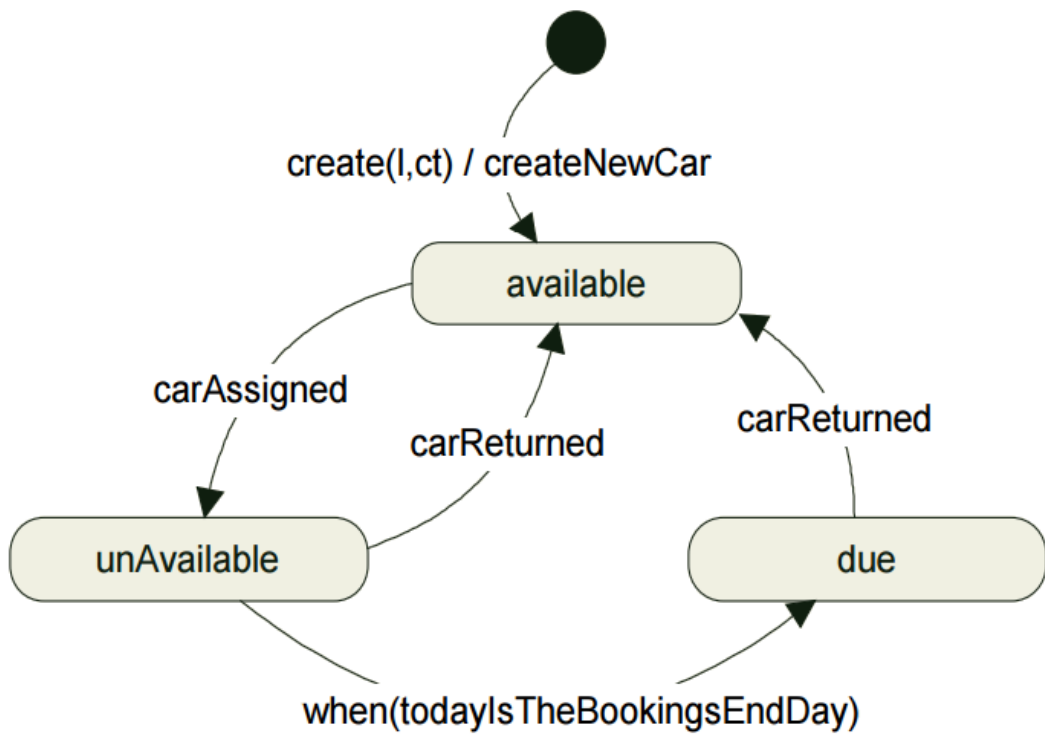


Fig 3.5 Textual Details of car Booking

3.2 Let Us Sum Up

In this case study we see that first the use case diagram shows overview on the system, its boundaries and its main functionality. Secondly, object diagrams and sequence diagrams are created as required scenarios for system structure and behaviour. After this, develop a class diagram and a statechart diagram for each class along with textual description of statechart diagrams with formal description for statecharts.

Then, check for existing object diagrams and sequence diagrams consistency with class diagram and state chart diagrams and modify if required. Finally, when operations is stable, start to develop activity diagrams for particular operation that start with textual description and advance to more formal one.

3.4 Glossary

1. **Class diagram** - An arguably foremost UML diagram type serving as building block of objects oriented answer showing classes in system, attributes and class operations.
2. **Object diagram** - Similar to class diagrams showing relationship among objects in real world that describes about system appearance at particular time.
3. **Activity diagram** - The model showing behaviour of system where behaviours are connected in overall flow of system.

3.5 Assignment

Explain the customer case diagram.

3.6 Activities

Explain the details about use case book from above case study?

3.7 Case Study

Compile and run the textual details about booking state chart.

3.8 Further Readings

1. Schulte, J. Niere: Thinking in Object Structures: Teaching Modelling in Secondary Schools; in Sixth Workshop on Pedagogies and Tools for Learning Object Oriented Concepts, ECOOP, Malaga, Spain, 2002
2. Zündorf: Rigorous Object Oriented Software Development, Habilitation Thesis, University of Paderborn, 2001.

Block Summary

In this block, you will understand about the basic of Functional Modelling and its link with object oriented analysis model that describes internal processes of system. The block gives an idea on architecture and distribution of details regarding Data Flow Diagrams. The examples related to concept of state diagram and characteristics are discussed.

In this block, you will understand about the features of Unified Modeling Language. The concept related to Data flow diagram showing graphical arrangement of information along with data movement steps is well detailed. You will be demonstrated practically about object diagram which is same as class diagrams with relationship in real world.

Block Assignment

Short Answer Questions

1. What are the features of Functional model?
2. What are Class diagrams?
3. What are the advantages and drawbacks of Data flow diagram?
4. What are Object and dynamic model?
5. Explain the concept of Behavioral Diagrams?

Long Answer Questions

1. Explain about Object Diagram?
2. What are Sequence diagram?
3. Explain about UML?

Enrolment No.

1. How many hours did you need for studying the units?

Unit No	1	2	3	4
Nos of Hrs				

2. Please give your reactions to the following items based on your reading of the block:

Items	Excellent	Very Good	Good	Poor	Give specific example if any
Presentation Quality	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	_____
Language and Style	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	_____
Illustration used (Diagram, tables etc)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	_____
Conceptual Clarity	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	_____
Check your progress Quest	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	_____
Feed back to CYP Question	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	_____

3. Any Other Comments

.....

.....

.....

.....

.....

.....

.....

.....



“

*Education is something
which ought to be
brought within
the reach of every one.*

”

- Dr. B. R. Ambedkar



Dr. Babasaheb Ambedkar Open University
'Jyotirmay Parisar', Opp. Shri Balaji Temple, Sarkhej-Gandhinagar Highway, Chharodi,
Ahmedabad-382 481.